

FOSTERING TECHNICALLY AUGMENTED HUMAN COLLECTIVE INTELLIGENCE



With an application to
human fluency in formal languages
for automated deduction

Chide Groenouwe

Fostering technically augmented human
collective intelligence
With an application to human fluency in formal
languages for automated deduction

Chide Groenouwe

read online



https://bit.do/col_int_cg



SIKS Dissertation Series No. 2019-20

The research reported in this thesis has been carried out under the auspices of SIKS, the Dutch Research School for Information and Knowledge Systems.

Cover art: Ebere Groenouwe – www.eberegroenouwe.nl

Cover layout: Jesse Nortier – www.jordgubb.design

The symbol on the cover has been inspired by the vein pattern of the Victoria Amazonica (an Amazonian water lily). It symbolises several aspects of collective intelligence, amongst which its recursive structure and its interconnectedness. It is the logo of the Nanquanu Institute for Collective Intelligence, with which the author is affiliated:

www.nanquanu.org

Copyright © 2019 Chide Groenouwe

ISBN 978-90-393-7162-6

Fostering technically augmented human collective intelligence

**With an application to human fluency in formal languages for
automated deduction**

Bevordering van technologisch verrijkte menselijke collectieve intelligentie

Met een toepassing op menselijke vloeiendheid in formele talen voor
automatische deductie
(met een samenvatting in het Nederlands)

Proefschrift

ter verkrijging van de graad van doctor aan de

Universiteit Utrecht

op gezag van de
rector magnificus, prof. dr. H.R.B.M. Kummeling,
ingevolge het besluit van het college voor promoties
in het openbaar te verdedigen op

maandag 8 juli 2019 des middags te 12.45 uur

door

Chidi Nwaneri Groenouwe

geboren op 10 oktober 1974 te Utrecht

Promotor: **Prof. dr. J-J.Ch. Meyer**

Call to the reader for feedback

Every book of a considerable size is prone to editing, typing and language errors. If you, as a reader, encounter such errors, I will be very grateful if you communicate these to me, the author. Even if you do so many years after the completion of this book. The corrections will find their way into the subsequent versions of this book. You can send your feedback to:

`info@nanquanu.org`

Your message will be processed by me or one of the contributors of the Nanquanu Institute. If the contact information is outdated, please download the newest version of this book at:

`https://bit.do/col_int_cg`

I appreciate it if you check whether the error already has been fixed in the newest version of this book.

Chide Groenouwe

Summary

This study addresses a general concern, a specific concern, and a synthesis between them. The general concern is how to foster human collective intelligence, in this work approached from a human-technological angle. The specific concern is how to foster human fluency in formal languages for automated deduction.

Human collective intelligence occurs, loosely speaking, when the intelligence of a group of people is greater than the sum of the intelligence of the individuals of that group. Important motivations to foster human collective intelligence are to help people (1) deal better with problems and (2) become better, more fluent and more refined at what they are already good at.

It may be clear that fostering human collective intelligence is a highly interdisciplinary quest, with numerous facets to address. These can impossibly all be covered in detail in one study. Rather, this study covers a large class of facets with an abstract overarching approach, and then works out a limited, yet carefully selected set of facets in detail.

The overarching approach (the Orcoba-Approach) facilitates human individuals or collectives to acquire or improve a given capability. The core of the approach consists of expressing the capability in a permanent record that enables the targeted capability to be, at least partially reproduced and examined. Such a record contains both a description of relevant human capabilities and activities, and specifications of complementary artefacts used. For example, a record of a team of firemen may contain English texts describing procedures followed while extinguishing fire, instruction movies used for training firemen, and a detailed specification of fire hoses and other equipment. The approach facilitates people, or other actors, to share, add and adapt records, and does so guided by the differences in performance teams achieve with them. It does so in a way inspired by biological evolution, accelerating the improvement of the best records.

The specialisations and implementations of the overarching approach developed in this study focus both with respect to the means used and the challenges ad-

dressed. The focus with respect to the means is on a complementary set of advanced computer technologies, human capabilities and human activity (human-computer-technological systems). Therefore, this focus may appeal most to those interested in integrating insights and techniques associated with computer science, social sciences and humanities. (Associated keywords: technologically augmented collective intelligence, hybrid artificial intelligence (evolutionary computation), technologically enhanced learning, computer supported collaborative work, organisational science, human-technological systems, human computation, crowd-sourcing, serious games, e-humanities). Equipped with these means, this study focuses on some major subchallenges within the overarching approach. Most subchallenges relate to an effective implementation of the evolutionary-inspired process of improving the mentioned records. This implementation turns out to be far from trivial. The challenges include: (1) How to deal with resource scarcity? Among other things, the effort of the collective of developers of records is limited. How to effectively distribute this effort over the records? (2) How to foster human motivation? (3) How to quantify the performance of humans using the record? This is a crucial instrument to determine the quality of the records. (4) How to deal with capabilities that require complex and large records? Among other things, the effect of a change in a record will then, in general, be difficult to measure. The longer a record, the smaller is the change with respect to the complete record. Hence, in general, the effect of a change on the performance of the users will also be smaller.

The questions are mostly addressed by presenting a family of detailed specialised designs of the overarching approach, a partial implementation of the software-aspect of these designs and experimentation with them.

The concern for fostering human fluency in formal languages for automated deduction (briefly: formal fluency) is inspired by the following. Since the second half of the 20th century much work has been done on developing and improving the application of computers to boost our capability to reason over our collective knowledge. One major branch of development involves automated deduction: knowledge that is represented in a formal language for automated deduction allow special computer programs to automatically produce knowledge-representations that are deducible from the given knowledge, regardless of by whom, where and why each part of the representation was created. (Associated keywords include: Semantic Web, Knowledge Representation and Reasoning, Knowledge Engineering, Ontology Engineering and Automated Theorem Proving.) This technique enables people to extend works of others to an unprecedented degree, contributing to the general concern of this work to increase human collective intelligence.

Although automated deduction is nowadays widely applied for the purpose just given, we are only scratching the surface of its full potential. One important challenge, among others, is that the amount of knowledge represented in a formal language for

automated deduction, while in absolute sense seemingly impressive, only represents the tiniest fraction of the ever-growing body of information produced by humans and their technologies. A way to accelerate the creation of these representations is to foster more people reaching formal fluency. The latter is the specific concern of this study.

As the reader may have guessed, this work's contribution to fostering formal fluency consists of subjecting it to the specialised approaches to foster human collective intelligence developed in the first part of the work. This work presents both how to tailor these approaches to formal fluency and the lessons learnt about favourable elements for a record for formal fluency. The most sophisticated fruit of this work consists of the design and partial implementation of the serious game developed for this purpose, SWiFT.

Samenvatting

Deze studie richt zich op een algemeen en een specifiek aandachtsgebied, en een synthese van beide. Het algemene aandachtsgebied is hoe technologisch verrijkte menselijke collectieve intelligentie te bevorderen, in dit werk benaderd vanuit een mens-technologische hoek. Het specifieke aandachtsgebied is hoe zogenaamde menselijke vloeiendheid in formele talen voor automatische deductie te bevorderen.

Er is grofweg sprake van menselijke collectieve intelligentie, indien de intelligentie van een groep mensen groter is dan de som van de intelligentie van de individuen uit die groep. Belangrijke redenen om menselijke collectieve intelligentie te bevorderen is om mensen te helpen (1) beter problemen aan te kunnen pakken en (2) beter, vloeiender en verfijnder te worden in waar ze al goed in zijn.

Het zal duidelijk zijn dat het bevorderen van menselijke collectieve intelligentie een uiterst interdisciplinaire aangelegenheid is, met talloze facetten. Deze kunnen onmogelijk in detail worden meegenomen in één studie. In plaats daarvan behandelt deze studie een grote klasse van facetten met een abstracte overkoepelende methode, en werkt vervolgens een beperkte, maar zorgvuldig uitgekozen selectie van facetten uit in detail.

De overkoepelende methode (de Orcoba-Approach) faciliteert menselijke individuen of groepen in het verwerven of verbeteren van een gegeven capaciteit. De kern van de methode bestaat uit het vastleggen van informatie over de capaciteit in een permanent *record* dat het mogelijk maakt om de betreffende capaciteit op zijn minst gedeeltelijk te reproduceren en onderzoeken. Het record bevat zowel een beschrijving van de relevante menselijke capaciteiten en activiteiten, als specificaties van de aanvullende toegepaste artefacten. Bijvoorbeeld, bij een team van brandweerlieden zou het record kunnen bestaan uit teksten die beschrijven welke procedures gevolgd moeten worden bij het blussen een brand, instructie-films die gebruikt worden voor het trainen van brandweerlieden, en een gedetailleerde specificatie van de brandweerslangen en andere uitrusting. De methode faciliteert mensen en andere actoren om records te delen, toe te voegen en aan te passen, en doet dat op basis van het

verschil in de prestaties die de teams leveren met behulp van de records. Het doet dat op een wijze die geïnspireerd is door biologische evolutie. Zodoende versnelt de methode het verbeteringsproces van de records.

De specialisaties en implementaties van de overkoepelende methode die in deze studie zijn ontwikkeld, zijn gebaseerd op een selectie van zowel middelen als uitdagingen. Wat betreft de middelen richt de studie zich op een complementaire set van geavanceerde computer technologieën, menselijke capaciteiten en menselijke activiteiten (mens-computer-technologische systemen). Deze selectie zal vooral lezers aanspreken die geïnteresseerd zijn in het integreren van inzichten en technieken die gerelateerd zijn aan de informatica, sociale wetenschappen en geesteswetenschappen.

Met deze middelen richt deze studie zich op een aantal belangrijke deel-uitdagingen binnen de overkoepelende methode. De meeste deel-uitdagingen zijn gerelateerd aan de effectieve implementatie van het evolutie-geïnspireerde verbeteringsproces van de genoemde records. Deze implementatie blijkt verre van triviaal. Uitdagingen zijn, onder andere: (1) Hoe om te gaan met schaarsheid aan middelen? Onder andere is het collectieve werk-potentieel van mensen die records ontwikkelen gelimiteerd. Hoe te zorgen voor een effectieve distributie van dit potentieel over de records? (2) Hoe mensen te motiveren? (3) Hoe de prestatie van mensen te meten die de records gebruiken? Dit is cruciaal om te bepalen hoe goed de records functioneren. (4) Hoe om te gaan met menselijke capaciteiten waarvoor lange en complexe records nodig zijn? Onder andere is het effect van een wijziging dan, in de regel, moeilijker te meten. Hoe langer een record, hoe kleiner de wijziging is in verhouding tot het totale record. Daarmee zal, in de regel, ook het effect op de prestaties van de gebruikers van het record kleiner zijn.

Op deze vragen wordt voornamelijk ingegaan door het presenteren van een familie van gedetailleerde specialiseerde ontwerpen van de overkoepelende methode, een partiële implementatie van het software-aspect van deze ontwerpen, en experimentatie met deze ontwerpen.

Het aandachtsgebied 'menselijke vloeiendheid in formele talen voor automatische deductie' (kortweg: formele vloeiendheid) is geïnspireerd door het volgende. Sinds de tweede helft van de 20^{ste} eeuw is veel werk verricht om computers toe te kunnen passen op het vergroten van ons vermogen om te redeneren over onze collectieve kennis. Een belangrijke tak van deze ontwikkeling betreft *automatische deductie*: kennis die gerepresenteerd is in een formele taal voor automatische deductie staat speciale computer programma's toe om automatisch nieuwe kennis-representaties te produceren die deduceerbaar zijn uit de gegeven representaties, ongeacht door wie, waar en waarom elk deel van de gegeven representaties werd gecreëerd. Deze techniek stelt mensen in staat om in een voorheen ongeëvenaarde mate voort te bouwen op het werk van anderen. Op deze wijze draagt het ook bij aan het algemene aandachtsgebied van dit werk om technologisch verrijkte menselijke collectieve

intelligentie te bevorderen.

Ondanks het gegeven dat automatische deductie vandaag de dag breed wordt toegepast voor het zojuist genoemde doel, benutten we nog maar een fractie van haar volledige potentie. Een belangrijke uitdaging is, onder andere, dat de hoeveelheid kennis die gerepresenteerd wordt in formele talen voor automatische deductie slechts een fractie vormt van de voortdurend groeiende hoeveelheid informatie die geproduceerd wordt door mensen en hun technologieën. Een manier om de creatie van deze representaties te versnellen is het bevorderen van formele vloeiendheid. Dit is het specifieke aandachtsgebied van deze studie.

De lezer heeft wellicht al geraden dat de bijdrage van dit werk aan het bevorderen van formele vloeiendheid bestaat uit het toepassen van de eerder genoemde gespecialiseerde methoden ter bevordering van menselijke collectieve intelligentie op formele vloeiendheid. Het werk presenteert (1) hoe deze benaderingen op maat gemaakt kunnen worden voor formele vloeiendheid en (2) de geleerde lessen ten aanzien van gunstige ingrediënten van een record voor formele vloeiendheid. De meest verfijnde vrucht van dit werk bestaat uit het ontwerp en de gedeeltelijke implementatie van een *serious game* voor dit doel: SWiFT.

x

For Marre (Marijke), whose love and wisdom made me who I am today.

Contents

Call to the reader for feedback	i
Summary	iii
Samenvatting	vii
I General	1
I.1 Introduction	3
I.1.1 Targeted reader audiences	4
I.1.2 Structure and notation	5
I.1.2.1 Research questions and results	5
I.1.2.2 Fixed terminology for concepts	6
I.1.2.3 Clickable links in PDF readers	7
I.1.3 Overall research questions	7
I.1.4 Approaches	8
I.1.4.1 Hucolligence	8
I.1.4.2 Formal-fluency	11
I.1.5 Targeted reader audiences (continued)	12
I.1.5.1 Computer scientists, exact scientists and technologists . .	12
I.1.5.2 Social sciences and humanities	14
I.1.6 Overview and structure	16
II Fostering human·collective·intelligence	19
II.1 Overview and structure	21
II.2 Introduction	23

II.2.1	What is human-collective-intelligence?	23
II.2.1.1	<i>Intelligence</i>	26
II.2.1.2	<i>Human-collective-intelligence</i>	27
II.2.1.3	Related research	28
II.2.1.4	Dimensions	39
II.2.1.5	The transcendental dimension	40
II.2.1.6	Fostering hucolligence	41
II.2.1.7	Fostering co-evolution	41
II.2.2	Research questions	42
II.3	The Orcoba-Approach	43
II.3.1	The human and artefact part	44
II.3.2	Constitutional-aspects and contributive-aspects	44
II.3.3	The human part of a capability-constitution	45
II.3.4	The artefact part of a capability-constitution	46
II.3.5	Examinability of composition-records	46
II.3.6	Openness of Cobas	46
II.3.7	Application areas	47
II.3.8	Research question refinement	47
II.4	Orcoba-extensions and generalisations	49
II.4.1	Introduction and research question refinement	49
II.4.2	Classes-of-systems and extensions	52
II.4.3	System-extension overview diagrams	53
II.4.4	Extensions and generalisations	54
II.4.5	Challenge-Based-Coba	57
II.4.5.1	Challenges and challenge-sets	57
II.4.5.2	Challenge-assessment	61
II.4.5.3	Designing challenge-generation and assessment systems	61
II.4.6	Higher-order-Orcobas	61
II.4.7	Conclusion	62
II.5	Extensions inspired by evolution theory	65
II.5.1	Overview and structure	65
II.5.2	Introduction and research questions	66
II.5.3	Introduction to evolutionary computation	67
II.5.3.1	Definitions and properties	68
II.5.3.2	Design considerations	70
II.5.4	Definitions from general evolutionary computation	71

II.5.5	Existing techniques within evolutionary computation	73
II.5.5.1	Proportional selection operator: stochastic universal sampling	74
II.5.5.2	Selective fitness function: Sigma scaling	75
II.5.6	Evolutionary human-technological systems (EvoHuts)	76
II.5.6.1	Orcoba systems and human-capability-enhancing-EvoHuts	76
II.5.6.2	Types of objective fitness functions in EvoHuts	77
II.5.6.3	Performance-index candidates	82
II.5.6.4	Stochastically measured dynamic performance	87
II.5.6.5	Relevant statistical theory	91
II.5.6.6	Curve fitting and regression analysis	99
II.5.6.7	Statistical estimation of the performance-function	103
II.5.6.8	Flatline-estimation	103
II.5.6.9	Applying sequential analysis	104
II.5.6.10	Growth-estimation	106
II.5.6.11	Joining growth-estimation and flatline-estimation	106
II.5.6.12	Future work join-graph	107
II.5.6.13	Future work on performance function estimation	108
II.5.7	Or-evohut	110
II.5.7.1	Or-evohut branches	111
II.5.7.2	Iteration 0 (initialisation)	112
II.5.7.3	Iteration 1 (expansion)	113
II.5.7.4	Iterations $i \geq 2$	113
II.5.8	Dealing with players who have trouble with a challenge	114
II.5.8.1	Design ingredient panic button	115
II.5.8.2	Design ingredient time-out studying time	115
II.5.8.3	Negative side effects of the panic button	115
II.5.9	Conclusion and future work	116
II.5.9.1	Conclusion	116
II.5.9.2	Future work	121
II.6	Conclusion and future work	125
II.6.1	Conclusion	125
II.6.2	Future work	126
II.6.2.1	Or-evohut	126
II.6.2.2	Higher-order-Orcobas	127

III	Fostering formal-fluency	129
III.1	Overview and structure	131
III.2	Introduction	133
III.2.1	Formal-fluency	133
III.2.1.1	‘Fluency’ in natural languages	134
III.2.1.2	‘Fluency’ in formal-languages	134
III.2.1.3	Why foster formal-fluency?	136
III.2.2	History and nature of automated-deduction	136
III.2.3	Amplifying human reasoning with automated-deduction	138
III.2.4	Terminology	141
III.2.4.1	Existing formal-languages	141
III.2.4.2	KR-base	142
III.2.5	Research question refinement	142
III.2.6	Feasibility	143
III.2.6.1	Small scale adoption as breeding place	145
III.2.7	Related work	145
III.2.7.1	Systems to train formal-fluency	145
III.2.7.2	The Semantic Web	145
III.2.7.3	Controlled natural languages	146
III.2.7.4	Semantic Wikis and related systems	147
III.2.7.5	Psycholinguistics and neuroscience	147
III.3	Integration of automated-deduction into society	151
III.3.1	Introduction	151
III.3.1.1	Why a measurable definition matters	152
III.3.1.2	Current status	153
III.3.2	Reflections on fostering desired states of affairs	154
III.3.2.1	Determining the right scope	154
III.3.2.2	Trade-offs in defining and applying metrics	156
III.3.2.3	Control	158
III.3.2.4	Resource limitations	158
III.3.2.5	Decentralised control	159
III.3.2.6	Self-similar state of affairs	159
III.3.2.7	Commitment	160
III.3.2.8	Trade-offs in pursuing states of affairs	161
III.3.3	Approach	162
III.3.3.1	Structure of the survey	162
III.3.3.2	Notations	164

III.3.3.3	Tip of the iceberg	164
III.3.4	Preliminaries	165
III.3.4.1	Overall metric: metric-for-integration-of-automated-deduction	165
III.3.4.2	The choice list of relevant-assumptions	170
III.3.5	Focus-based challenge analysis	172
III.3.5.1	Fostering integrations-of-automated-deduction under Ideal assumptions	172
III.3.5.2	Metric: automatically-deducible-fraction	174
III.3.5.3	Translation prioritisation	179
III.3.5.4	Trade-off expressivity and complexity	180
III.3.5.5	Fragmentation of formal-languages	184
III.3.5.6	Formal-languages as sets of semantic conditions	186
III.3.5.7	Efficiency of formal-language representations	187
III.3.5.8	Selection of automated-deductive-reasoners	189
III.3.5.9	Further areas of focus	189
III.3.5.10	Balancing the trade-off between quality and speed	190
III.3.5.11	Evolution of knowledge models	190
III.3.5.12	Dealing with rapid changes in information	190
III.3.5.13	Development of new automated-deduction techniques	191
III.3.5.14	Trust	191
III.3.5.15	Inconsistencies	191
III.3.5.16	Privacy and security	192
III.3.5.17	Future work	192
III.4	Coba extensions for fostering formal-fluency	193
III.4.1	Introduction	193
III.4.2	Compatibility	193
III.4.3	Extensions	194
III.5	Real-time collective formal-thinking	197
III.5.1	Introduction	197
III.5.2	Related work	199
III.5.2.1	Overview	201
III.5.3	Initial composition-record	201
III.5.3.1	Collaboration rules	201
III.5.4	The formal-language	202
III.5.4.1	Exact-Entity-Relation-Graphs (EERGs)	202
III.5.5	Core properties of entity-definitions	206
III.5.6	Discussion	208
III.5.6.1	Improving focus	208

III.5.6.2	Changes to the contributive-aspects	209
III.5.6.3	Changes to the constitutional-aspects of the method	210
III.5.7	Future work	210
III.5.7.1	Formal-languages with higher expressivity	211
III.5.7.2	Version management	211
III.5.7.3	Further integration of the Orcoba-Approach	211
III.5.7.4	Coordination of the real-time-collective-formal-thinking process	212
III.5.7.5	Decentralised development of personalised educational nodes	213
III.6	The SWiFT-class-of-games	215
III.6.1	Introduction	215
III.6.2	Related work	215
III.6.2.1	Games with a purpose	215
III.6.2.2	Translating natural language texts	216
III.6.2.3	Collaborative knowledge creation	216
III.6.3	SWiFT-Base	217
III.6.4	Validation of design decisions	218
III.6.4.1	The base metric for the quality of translations	218
III.6.4.2	Queries as algorithms	219
III.6.5	Extensions and variants	220
III.6.5.1	SWiFT-Full	220
III.6.5.2	Sub-research-questions	221
III.6.5.3	Complementary approach	222
III.6.5.4	Iterative approach	222
III.6.5.5	SWiFT-Fixtal	223
III.6.5.6	Higher-order-SWiFT and SWiFT-Focused	223
III.7	SWiFT-Fixtal and SWiFT-Fixtal-α	225
III.7.1	Introduction	225
III.7.2	Formal-target-language and reasoners	226
III.7.3	The composition-record	227
III.7.3.1	Overview of the composition-record	227
III.7.3.2	Formal-target-language	229
III.7.4	The game design	235
III.7.4.1	Game phases	235
III.7.4.2	Measuring the quality of translations	238
III.7.5	Remedies violations node definitions based on experiment	239
III.7.6	Conclusion	240
III.7.6.1	Fostering hucolligence with Orcobas	240

III.7.6.2	Fostering formal-fluency on a meta-level	242
III.7.6.3	Fostering formal-fluency on an object-level	243
III.8	SWiFT-Focused	245
III.8.1	Introduction and research questions	245
III.8.1.1	Research question refinement	246
III.8.1.2	Introduction solutions	246
III.8.1.3	Implementation of SWiFT-Focused	248
III.8.2	SWiFT-Focused rounds in detail	249
III.8.2.1	Consti-study-round	249
III.8.2.2	Translation-round	250
III.8.2.3	Bridge-construction-round	250
III.8.2.4	Queries as algorithms	250
III.8.2.5	Question-attack-round	255
III.8.2.6	Algorithmic-defence-round	256
III.8.3	Performance metric options	259
III.8.3.1	Future work	261
III.8.4	SWiFT-Focused design process and template	262
III.8.5	The formal-languages	264
III.8.5.1	General definitions and terminology	264
III.8.5.2	First-order logic	266
III.8.5.3	Efe-language	270
III.8.5.4	Folsequa-language	270
III.8.5.5	Pattern-languages Plosequa-language and Plosequamo-language	272
III.8.5.6	Folminquon-language	274
III.8.5.7	Folnuminquon-language	276
III.8.5.8	Pattern-languages Plonu-language and Plonumo-language	278
III.8.5.9	Implementation of a reasoner for Plonumo-language	280
III.8.5.10	Bridge-language Basic-language	286
III.8.5.11	Hurelan-language	288
III.8.6	SWiFT-Efe	288
III.8.6.1	The challenge-set	289
III.8.6.2	Formal-languages	289
III.8.6.3	The question set	290
III.8.6.4	Typical session example	290
III.8.7	SWiFT-NoUna	292
III.8.7.1	The challenge-set	293
III.8.7.2	Formal-languages	293

III.8.7.3	The question set	294
III.8.7.4	Typical session example	294
III.8.8	Conclusion and future work	296
III.8.8.1	Conclusion	296
III.8.8.2	Future work	301
III.9	Conclusion and future work	303
III.9.1	Conclusion	303
III.9.2	Future work	307
III.9.2.1	SWiFT	307
III.9.2.2	Transforming Wikipedia	308
III.9.2.3	Real-time-collective-formal-thinking	310
IV	Experimentation	311
IV.1	Overview and structure	313
IV.2	System-Real-α	315
IV.2.1	Introduction	315
IV.2.2	Goal	315
IV.2.3	Results of experiments	316
IV.2.3.1	Quality of defined entities	317
IV.2.3.2	Core properties	317
IV.2.4	Conclusion	320
IV.2.4.1	Performance	320
IV.2.4.2	Reducing human fallacies	320
IV.2.4.3	Changes to the essence of the method	321
IV.2.4.4	Changing focus	322
IV.2.5	Future experimentation	322
IV.3	SWiFT-Fixtal-α	323
IV.3.1	Introduction	323
IV.3.1.1	Game setting during experiment	323
IV.3.1.2	Additional elements of the experimental setting	325
IV.3.2	Results and analysis	326
IV.3.2.1	Score	326
IV.3.2.2	Question-attack-round	326
IV.3.2.3	Violations guidelines node definitions	327
IV.3.2.4	Motivation	330

<i>CONTENTS</i>	xix
V Conclusion and future Work	333
V.1 Conclusion	335
V.2 Future work	339
V.2.1 Fostering hucolligence with Orcobas	339
V.2.2 Fostering formal-fluency	339
V.2.2.1 SWiFT	339
V.2.2.2 Real-time-collective-formal-thinking	340
Afterword	341
V.2.3 Acknowledgements	345
Glossary	349
Curriculum vitae	367
Bibliography	385
SIKS dissertation series	387

Part I

General

Chapter I.1

Introduction

This book addresses two related central concerns, one general and one specific. The general concern is how to foster human collective intelligence. The specific concern is how to foster human fluency in formal languages for automated deduction. The following first briefly characterises and motivates both, and then sketches their relationship.

Human collective intelligence cannot be defined, it can only be characterised. The following imprecise, but illustrative description, suffices for this introduction: human collective intelligence occurs when the intelligence of a group of people is greater than the sum of the intelligence of the individuals of that group. An important motivation for fostering human collective intelligence is to help people solve problems. This is an urgent matter, given the current incapability of humankind to cope with the exceedingly complex consequences of their own (technological) inventions. This is, among other things, evidenced by humankind's tremendous difficulty to establish a sustainable way of living. Douglas Engelbart was one of the first driven by this motivation and provided a modern definition of the field and launched an extremely successful program in the early sixties (Engelbart, 1962). Another major motivation is to help people become better in what they are already good at. In other words, to feed the desire of a continued growth in insight and capability, pursuit of harmony with ourselves and the world surrounding us and to express ourselves. The latter is a perennial drive, remaining applicable even if we would find a way to overcome the most pressing problematic complexities.

The concern for fostering human fluency in formal languages for automated deduction (briefly: formal fluency) is inspired by the following. Since the second half of the 20th century much work has been done on developing and improving the application of computers to boost our capability to reason over our collective

knowledge. One major branch of development involves automated deduction: knowledge that is represented in a formal language for automated deduction enable special computer programs to automatically produce knowledge-representations that are deducible from the given knowledge, regardless of by whom, where and why each part of the representation was created. (Associated keywords include: Semantic Web, Knowledge Representation and Reasoning, Knowledge Engineering, Ontology Engineering, Automated Theorem Proving). This technique allows people to extend works of others to an unprecedented degree, contributing to the general concern of this work to increase human collective intelligence.

Automated deduction is nowadays widely applied for the purpose just given. Nevertheless, we are far from exploiting its full potential. An important challenge is that the amount of knowledge represented in a formal language for automated deduction, while in absolute sense seemingly impressive, only represents a fraction of the ever-growing body of information produced by humans and their technologies.

An approach to accelerate the creation of these representations is to foster an increased formal fluency among people. In the context of this work, I define formal fluency as the ability to fluently express information in a deductive formal language, where ‘fluency’ is used in a sense that is restricted to the dimensions speed, accuracy, effortlessness and the degree to which automated deduction can be effectively applied to the resulting expressions.

I.1.1 Targeted reader audiences

The topics addressed in this book are of an interdisciplinary, and even transdisciplinary nature. This requires the integration of instruments and goals that exist within several disciplines (‘inter’), and even beyond them (‘trans’). The philosophy that underlies this work is that the topic of collective intelligence requires such an integrative approach to be, ultimately, successful. Some topics introduced here may therefore not readily spark strong familiarity among readers that move *within* the boundaries of one of the established (sub)disciplines, in spite of the fact they may be among the targeted audiences of this work. Examples are computer science and sociology. It is clear that formal fluency is strongly related to computer science. However, how does fostering human collective intelligence relate to ground-breaking computer science research? The reverse may be asked for sociology. Consequently, it may be important to explicitly point out how this book is relevant and valuable for these readers. However, doing so, is most easily realised after some preliminary notions have been developed. Section I.1.5 (p. 12) aims to fulfill the promise made here. This introduction, therefore, does a kind appeal to the patience of these readers.

I.1.2 Structure and notation

This section defines the structural and notational elements that are used in the rest of this book.

I.1.2.1 Research questions and results

This work is systematically organised in a hierarchy of *research questions*. They form an important back-bone of this work.

I.1.2.1.1 Hierarchical organisation

Each research question is assigned a dot-separated sequence of numbers. This sequence indicates its position in the hierarchy of related research questions. It does so as follows. Each top-level research question is numbered with a unique single number. Each sub-research question is numbered with the number of its ‘parent’-research question, followed by a dot and then again, a unique number. For example:

Research question 1. [*Example of a main research question*] *How to organise research questions effectively?*

[*Conclusion(s): [page with result]*] ■

indicates that research question 1 is a top-level research question. The following question is an example of the first sub-research question of research question 1:

Research question 1.1. [*Example of a sub-research question*] *How to organise sub-research questions effectively?*

[*Conclusion(s): [page with result]*] ■

Hence, note that the questions are *not* numbered according to the section in which they occur, they are numbered with relation to each other.

I.1.2.1.2 Navigation

Each research question refers to a summary of the result of its investigation, and the summary, in turn, refers back to its research question. Additionally, each result refers to the sub-research questions of its research question, if there is any. In the digital version of this book, there are also hyperlinks provided, which allows easy navigation between corresponding research questions, sub-research questions and results. Note that the *number* in the title of each research question contains a direct link to its parent research question. For example, in the aforementioned research

question 1.1, the ‘1’ is hyperlinked, and by clicking it, the reader will navigate to research question 1. (It may be clear that if both sub-research question and parent research question occur on the same page, nothing will happen.)

I.1.2.1.3 Locations

Sub-research-questions are typically introduced only after sufficient material has been developed that gives rise to them. This means that a sub-research question may be located a few, to many pages after its parent question.

I.1.2.2 Fixed terminology for concepts

The following holds for terminology introduced and used in this book.

- For a selection of concepts important to this work, a fixed mini-phrase is introduced, that describes the concept briefly. (Typically, this is a noun phrase.) Examples are:
 - human·collective·intelligence
 - community·pursuing·a·capability

To disambiguate fixed mini-phrases from ordinary phrases, spaces in it are replaced with dots. The latter prevents misunderstandings: it indicates that the mini-phrase has a specific, explicitly defined meaning within the context of this book.

- Fixed mini-phrases that occur often throughout the work are abbreviated with a word that is pleasantly legible and pronounceable, and still resemble the expanded form to such a degree that is, hopefully, easier to remember. For example human·collective·intelligence is abbreviated with hucolligence. An additional advantage of such an acronym, is that it is a new word which prompts the reader that a notion has a meaning that is specific to the context of this book.
- Alternatively, fixed mini-phrases are abbreviated with a shorter word sequence with typically known words. For example, formal·language·for·automated·deduction is abbreviated with formal·language. The disadvantage of this is that the reader may interpret the abbreviated term incorrectly if the reader forgot the specific definition of the term in the scope of this book. In the case of the example: the term formal·language circulates with related, though slightly different meanings in the scientific community.

- If there is a large distance between two subsequent occurrences of the same abbreviation, or when an abbreviation occurs the first time in a larger unit (such as a chapter), it may be supplemented with its extended form to refresh the mind of the reader. For example: hucolligence (human·collective·intelligence).
- All abbreviations and mini-phrases are included in the glossary.
- Each first usage of a fixed phrase or abbreviation is hyperlinked to its entry in the glossary. Examples are in the previous points.
- To make it easy to find the definition of a concept, the glossary contains annotations: The page number that refers to the page where the term is defined (or described) most elaborately is typeset differently than the rest of the page numbers. An example: ‘110’ is an ordinary page reference, while ‘117’ refers to the page with the definition.

I.1.2.3 Clickable links in PDF readers

This book is enriched with a great number of clickable links that make it much easier to navigate it. Among other things, all cross-references to sections, definitions, figures, research questions, terms that occur in the glossary and references to the literature are clickable. It is important to note that there are PDF viewers that do not show these links. A test has revealed that they are shown in stand-alone desktop PDF readers, such as Adobe Acrobat Reader, but not in online PDF viewers that are integrated in web browsers or in e-readers. In e-readers the links, however, sometimes seem to work even if they are invisible.

If one wants to take optimal advantage of the navigation possibilities, I recommend using a stand-alone desktop PDF reader.

I.1.3 Overall research questions

The top-level research questions of this work are the following.

Research question 1. [*The general quest*] *How to foster human·collective·intelligence (or briefly: hucolligence)?*
[*Conclusion(s): page 335*] ■

In the sequel, human·collective·intelligence will mostly be abbreviated with *hucolligence*.

hucolligence

Research question 2. [*The specific quest*] *How to foster human·fluency·in·formal·languages·for·automated·deduction (or briefly: formal·fluency)?*
[*Conclusion(s): page 336*] ■

Table I.1.1: Human capabilities and complementary artefacts.

<i>Human capabilities</i>	<i>artefacts</i>
reading, writing	pen, paper, glasses
speaking and listening	telephone, Skype, microphone
formal-fluency	networked computer technology

I.1.4 Approaches

I.1.4.1 Hucolligence

For the purpose of introducing the general approach chosen in this book for fostering hucolligence, the following first introduces some terminology. Moreover, it will give a more quantifiable characterisation of what it means to increase hucolligence, allowing for a sharper formulation of the requirements that the approach should meet.

I.1.4.1.1 Definitions related to hucolligence

*community-
pursuing-a-
capability
pursued-
capability*

First, a *community-pursuing-a-capability* in the context of this book, is characterised as a community that pursues acquiring or improving a certain (collective) capability (its *pursued-capability*). *Examples: a team of firemen with the capability to collaboratively extinguish fire; a group of scientists collaboratively discovering new knowledge.* The capability may be *collective* in the sense that the capability emerges from the interaction between the members of the community. Associated words are: ‘collaborative’ and ‘communal’. A *capability-composition* in the context of this book, is the complete set of elements of which the given capability is composed. It is composed of human elements and complementary artefacts (tools). Examples are provided in table I.1.1.

*capability-
composition*

The *pursued-capability* also explicitly includes collective meta-capabilities. A collective meta-capability pertains to acquiring or improving other collective capabilities. Related notions are meta-cognitive skills, “learning to learn” and general intelligence. *Example: The capability to learn languages quickly, is a meta capability with respect to the capability of proficiency in French. The capability to learn anything (being general intelligence) is the most meta of all of them. Another, more popularly phrased meta-capability are ‘problem-solving skills’.* This work even encourages an inclusion of such meta-capabilities, because I associate them with a higher degree of hucolligence.

I.1.4.1.2 The Orcoba-Approach

The strategy for answering the question of how to foster hucolligence is by presenting a system and carrying out a validation of its effectiveness.

The system to be presented (the Orcoba-Approach) is inspired by the following basic principles:

1. Decentralised-development (later called: organisational self-similarity): instead of presenting an ‘object level’ composition to be imposed upon a community of collaborators, this work develops a ‘meta level’ system that promotes the emergence of communities within which each member participates in the construction and improvement of their community’s ‘object level’ composition.
2. Institutionalised-self-reflection: related to the previous point, the communities are, in addition to participating in the construction of the object-level composition, continuously and systematically reflecting on it as to effect its further growth.
3. Human-artefact-co-evolution: stimulation of a mutually responsive co-evolution of human capabilities and artefacts, i.e. both the dimension of human capabilities and the complementary dimension of artefacts are subjected to evolution in a mutually responsive way.

These points are motivated in the following way. First, in this study decentralisation is considered to be a crucial factor in fostering collective intelligence, as has also been explained by others (Visser and Berg, 1999; Visser, 2001; Levy, 1999; Engelbart, 1962; Coolen, 1992). Second, institutionalised self-reflection, a perpetual reflection by the members on their composition, is required to adapt the composition to match newly gained insights and new problems as closely as possible. Third, the following analogy may make it plausible that a mutually responsive co-evolution is a requirement for reaching the optimal composition-records: Suppose you try to maximise a function f from \mathbb{R}^2 to \mathbb{R} . Being limited to only vary one variable of f , while having to assume the other to be fixed, will in most cases not allow you to find a global maximum of f . The liberty of varying both variables, however, allows one to do so. The latter allows one to exploit the mutual *interaction* between both variables.

Another basic point that I consider to be of vital importance is the positive influence of cognitive diversity in collective intelligence. Page (2010) has done extensive research in this field. I have, for now, decided not to include it in the list of basic principles, because I consider it as a less universal point. Among other things, there are trade-offs associated with cognitive diversity, such as the time it takes to resolve disagreements in such settings, and the larger size of the communities needed. Hence, depending on available resources and goals a lesser or stronger degree of

cognitive diversity may be desirable. However, it can and should be included as an ingredient in composition-records. If it matters, these composition-records should outperform others in many cases.

Note that it is, due to limited research resources, beyond the scope of this book to rigorously validate the effectiveness of these principles. Rather, parts of the systems inspired by it, that is, more detailed extensions of the Orcoba-Approach, will be subjected to such an evaluation.

*Orcoba-
Approach
Orcoba
composition-re-
cord*

The *Orcoba-Approach* is defined as follows. *Orcoba* stands for Open Composition-Record Based Community. An Orcoba is a community-pursuing-a-capability, together with a record that allows the composition of the targeted capability to be (at least partially) reproduced and examined. The term *composition-record* is coined for the mentioned record. Ideally, the composition-record contains sub-records pertaining both to the artefacts and the human capabilities.

An example of such a record is that of a specific team of firemen. With regard to human capabilities this record may contain a document with procedures to collaborate while extinguishing fire and instruction movies used for training firemen. With regard to artefacts the record may contain a specification of tools used during the process of extinguishing fire, such as specific types of fire hoses.

Orcobas are (1) a concretisation of institutionalised self-reflection: a composition-record allows a composition to be examined, which allows an active reflection on, and therefore an improvement of it (point 1), and (2) in agreement with the idea to stimulate a fluent co-evolution of human capabilities and artefacts (point 3) (*end of list*).

The composition-record is, moreover, *open* in two senses: (1) it is open to the participants of the Orcoba, that is to say, they may participate in its construction and improvement, and (2) it is open for reuse by others (*end of list*). This is in agreement with decentralisation (point 2).

The Orcoba-Approach is defined and motivated in greater detail in chapter II.3 (p. 43). It serves as a base to explore several more detailed and concrete extensions of it, among which the serious game SWiFT. A central task of this book is to validate or repudiate the design decisions within the more detailed and concrete extensions of the Orcoba-Approach.

An important issue related to composition-records development is the trade-off between long-term and short-term performance. In other words: one composition-record may perform worse than another in the initial stage, but be better in the long run. More formally expressed: let R_1 and R_2 be composition-records for the same pursued-capability. R_1 has a better long-term performance than R_2 if for any given arbitrary infinite sequence of randomly chosen problems P_1, P_2, P_3, \dots that test the pursued-capability, and with which teams are faced successively, the average performance of the teams using R_1 catches up with the average performance of

teams using R_2 within a reasonable amount of time.

I.1.4.2 Formal-fluency

This study intends to contribute to the growth of formal-fluency in two general ways:

1. on a *meta level* by making plausible that specialised extensions of the Orcoba-Approach exist that can contribute to fostering (the emergence of good composition-records for) formal-fluency, and presenting these extensions, and
2. on an *object level* by presenting successful ingredients for composition-records that enable people to approach formal-fluency.

On the meta level, this book focuses on two types of extensions of the Orcoba-Approach: (1) extensions focusing on full formal-fluency, and (2) extensions focusing on isolated aspects of formal-fluency (*end of list*). The first type may seem sufficient. However, an excellent composition-record for full-blown formal-fluency is expected to be extensive and may take years to develop. In contrast, the second can be realised on much shorter time scales, but will not lead to a composition-record covering full formal-fluency. Moreover, it requires one to know in advance what problems could arise during practising formal-fluency, which can only be discovered by tests investigating full formal-fluency. The conclusion is that both methods should best be executed in parallel. The first type will help establish new isolated aspects of formal-fluency to address, and the second type may be used to subject these aspects to an accelerated, condensed, evolution. Moreover, given resource limitations of this study, considering the first type only would be infeasible. It is beyond the scope of this study to create a full implementation and validation of an Orcoba extension for fostering full formal-fluency. Rather, one goal is to make it plausible that such extensions exist, provide a prototype design, and validate the effectiveness of essential parts of it.

The meta level and object level are intended to interact. The application of an extension of the Orcoba-Approach (= a meta level process) is envisioned to lead to successful composition-records for formal-fluency (= object level products).

The specialised extensions of the Orcoba-Approach for the purpose of fostering formal-fluency are System-Real- α and SWiFT (Semantic Web in Fast Translation).

System-Real- α is a system that focuses on fostering formal-fluency as a sub-capability within *real-time-collective-formal-thinking*. In *real-time-collective-formal-thinking* a community collaboratively solves a problem, while all members express their thoughts about the problem at the moment they conceive of them in a fine-grained way in a persistent shared digital collective memory (a KR-base) using a

*real-time-
collective-
formal-thinking*

formal-language, according to a predefined set of rules. (A KR-base is a database with knowledge representations in a formal-language.)

SWiFT incorporates a competitive multi-player online game element into the Orcoba-Approach, among other things to motivate participants and maximise accessibility of SWiFT to many, accelerating the evolution process. SWiFT proceeds, in brief, as follows: Competing teams translate the same natural language text into a formal-language, each player covering a fragment that does not overlap with others of his team. Then teams challenge each other's translations by posing questions that have a very specific answer based on the content of the text (question attack). A question could for example be: give me all vitamins that act as a catalyst for cell growth. Each team then tries to construct a formal query that deduces the answer from their translation (algorithmic defence). A short translation time, a low amount of effort needed to construct the query, and correct answers contribute positively to the final score of a team. Several extensions and variants of SWiFT are explored in this book.

I.1.5 Targeted reader audiences (continued)

The preliminary notions are now sufficiently developed to clarify how this book is relevant and of use to readers from several contemporary disciplines, as promised in section I.1.1 (p. 4). The following clarifies this for several disciplines. It addresses the aforementioned main research questions separately.

I.1.5.1 Computer scientists, exact scientists and technologists

I.1.5.1.1 Fostering hucolligence

The approach chosen in this work for fostering hucolligence heavily draws on computer science, and more general, mathematical modelling and technology. It does so on two levels: on an *object* and a *meta* level.

The *object* level refers to the composition-records of a given Orcoba. After all, the composition-records are the objects that are generated or improved by means of the Orcoba-Approach.

The Orcoba-Approach encourages composition-records that incorporate advanced technology. This is in agreement with human-artefact-co-evolution. The scope of this book's study is limited to computer technology, because of its incredible and yet far from exhausted potential to enhance hucolligence. With this scope in mind, another way to phrase it is as follows. Where a major part of computer science and technology is about investigating and producing new *computer* programs, the quest of this book is to investigate and produce new 'programs' for *human-technological*

systems. Programs that, among other things, also contain aspects to effect certain human behaviours as part of the functioning of the system. This is the role of the aforementioned composition-records. Hence, these programs generalise the notion ‘computer program’. A computer program is now a special case of such a program: one that does not include human elements. Equivalently, it can be generalised to any technology, by replacing the word ‘computer program’ with ‘technological design’ in this paragraph.

The *meta* level refers to the form and design of the Orcoba-Approach and its extensions themselves. After all, the Orcoba-Approach and its extensions are the instruments that are used to foster the emergence of effective composition-records, its objects.

For the meta-level the same holds as for the object level: the design of the Orcoba-Approach integrates both technology and human capabilities as first-class citizens. Again, with regard to the technological component, the scope of this work is limited to computer science and computer technology. For example, the most elaborate extension has a design that draws inspiration from the field of Evolutionary Computation and statistics. Moreover, the implementation of the design consists for a large part of software. Typically, Orcobas are fully mediated by online software. (Compare this with for example Wikipedia, or computer supported collaborative work environments.) For example, I have launched and am the lead developer of a considerable open source project around one extension of the Orcoba-Approach. The project meets several professional standards for large software projects. It is written in one of the most advanced programming languages around – Scala¹ and several contributors have been involved.

At the time of writing, the project is available at the following location:

<https://github.com/swiftgame/SWiFTfososc>

In the near future, further development is to be expected.

I.1.5.1.2 Fostering formal-fluency

The object of research question 2 (p. 7), formal-fluency, self-evidently contains a strong computer science component, for formal-fluency is about improving effective and sensible application of automated-deductions to representations of information. Therefore, this work is in particular interesting for computer scientists from related fields such as knowledge representation and reasoning, automated theorem proving and for logicians. If one pursues an effective usage of automated-deductive-reasoners

¹Scala could be regarded as a variant of Java that elegantly integrates the functional programming paradigm.

in society, fostering complementary human capabilities, amongst which formal-fluency is indispensable. For the researcher with a transdisciplinary mindset, this in itself will be an interesting topic. However, this work is also interesting for those with an exclusive focus on the technological part. As has been shown over and over in the past, the application context of a discipline can have a tremendous influence on the research progress and directions *within* the boundaries of that discipline. For example, mathematics, even the pure branches, have been heavily influenced by questions asked in its application areas, amongst which physics and technology.

Glimpses of such influences are also manifest in this book. Among other things, mass participation requires a way to more easily create new formal-languages and reasoners. Moreover, this mass participation may lead to many more language variants coming into existence, and therefore a greater pool of research objects to test theory and practice regarding for example the trade-offs between reasoning complexity and expressivity of knowledge representation.

I.1.5.2 Social sciences and humanities

This section assumes social sciences and humanities to include disciplines such as economics, psychology, linguistics, educational science, organisation science and sociology.

I.1.5.2.1 Fostering hucolligence

It may be instantly clear that the object of research question 1 (p. 7), that is human-collective-intelligence, has, by definition, a strong social and human component. The words ‘human’ and ‘collective’ do not leave much to be guessed. Nevertheless, this work emphasises the use of advanced computer-scientific and mathematical instruments. This work, however, by no means intends to give the impression that it is based on the assumption that these are the only effective instruments for the given purpose. To the contrary, the overall quest is, as said, of a transdisciplinary nature, and should, therefore, be approached from many angles. The limitation in scope is primarily chosen because of the untapped huge potential of the integration of these technologies into hucolligence and the author’s background. More reflections on the transdisciplinary nature of the quest are to be found in chapter II.2 (p. 23). Many topics covered in this work, therefore, may be only be legible for a reader from one of the social sciences or humanities with a considerable mathematical and/or technological background. Nevertheless, there are also quite some topics that may directly appeal to social scientists without such a background. These readers are strongly encouraged to read the work, too. For example, the research questions in chapter II.4 (p. 49) contain many questions the answers of which can also be

approached from the perspective of the social sciences and humanities. Therefore, thoughts and views about this topic that complement the currently given answers are more than welcome.

The following covers some specific associations between this work and subdisciplines within social sciences and humanities. The coverage is not exhaustive, there may well many other parts of the work that appeal to these subdisciplines.

I.1.5.2.1.1 Psychology Psychologists may, for example, take interest in approaches to foster human motivation and the prevention of conservatism within Orcobas: for example, see the associated research questions in chapter II.4 (p. 49).

I.1.5.2.1.2 Educational science, educational design, and cognitive psychology Human learning is at the heart of this work. For example, chapter II.5 (p. 65) contains many topics that are of interest to, and could be further complemented with work from the educational scientist, designer and cognitive psychologist. This holds in particular for the method developed in section II.5.6 (p. 76) that proposes a way to systematically integrate empirical learning theory into Orcobas and related systems. This topic is, among other things, closely related to the following: how can we integrate empirical assessment techniques for learning materials in a crowd-source platform for learning material development?

I.1.5.2.1.3 Economics and organisation science For economists and organisational scientists it may, among other things, be interesting that the Orcoba-Approach develops a highly systematic human-technological method to effectively balance the trade-offs between exploration and exploitation in the process of determining how to invest human and other resources. This is the specific focus of chapter II.5 (p. 65).

I.1.5.2.2 Fostering formal-fluency

Though research question 2 (p. 7) (the quest to foster formal-fluency) is most directly associated with computer and exact sciences, it is also strongly interwoven with social sciences and humanities. It is interwoven in obvious, but also less obvious ways. First, it is obvious that formal-fluency requires human learning. In particular educational scientists and cognitive psychologists may take interest in the approaches and obstacles in this learning process. Second, it may also be obvious that fluency in a formal-language and fluency in a natural language are strongly related. Insight in both the differences and commonalities are invaluable to develop good composition-records for fostering formal-fluency. In particular the (psycho-)linguist may therefore take interest in the observations and conclusions drawn in this work, and is invited to think about how to complement these.

It may be less obvious that I deem it plausible that formal-fluency may influence the evolution of natural language. This does not mean that I blithely take the stance that such an evolution is a rapid and easy process. However, the ubiquitous integration of computer technology in our daily lives, already affects the way we think and communicate, and may, given enough time, also profoundly affect the way we use and learn language. If people discover that a specific way of expressing information within networked computer technology greatly improves collaboration, they may slowly, over time, acquire the associated capabilities. (Compare this with the invention and evolution of writing and reading, and its profound influence.) This work, in a sense, aims to contribute to the acceleration of this process. Hence, in the future formal-fluency may be integrated in the languages the majority of people use, perhaps even into parts of natural language itself. There already seem to be some indications for this in for example Twitter, in which people started using so-called *hashtags* to exploit the searching capabilities of algorithms. The use of hashtags has even made its entry into normal natural language usage beyond Twitter (Evans, 2015).

This would, in the future, turn the subject *by definition* into a topic covered by (psycho)linguists. Research efforts in this area may also partially work as a (desirable) self-fulfilling prophecy. If a sufficient amount of research energy is put effectively into lowering the gap between natural language usage and formal-language usage, this could accelerate the evolution towards the conjectured integration. Linguists are, therefore, strongly encouraged to read and complement this work. Moreover, the conjectured integration could also benefit from thorough reflection on it by the philosopher of language.

I.1.6 Overview and structure

This work is divided into three major parts:

- Part II (p. 21), which treats the first central concern of this book: fostering hucolligence by means of the Orcoba-Approach. It briefly covers the grand challenge of fostering hucolligence, and then proceeds with the focus of this work within this grand challenge: the Orcoba-Approach. It does so in much more detail than this section did so far, covering the overall approach as well as several extensions. One major reason for the introduction of extensions is that different applications of the Orcoba-Approach are better served by different specialisations of the base-system. The most elaborate general extension is inspired by evolution theory and evolutionary computation (see chapter II.5 (p. 65)).

- Part III (p. 131), which treats the second central concern of this book: fostering formal-fluency. The main motivation in this work for fostering formal-fluency, is that it is considered an important ingredient in the overarching challenge to improve the degree to which computers can assist people in reasoning about their knowledge, achieving unprecedented reasoning speed and reliability. This part of the book, therefore, also surveys some of the remaining challenges within this overall challenge, among other things to elucidate the place of formal-fluency within the latter (chapter III.3 (p. 151)). Subsequently, it defines several applications of Orcoba-systems to fostering formal-fluency, to wit: several SWiFT-game variants and System-Real- α .
- Part IV (p. 313), which treats experiments with some of the systems developed in the earlier parts. The main purpose is to validate design decisions in the light of their envisioned purpose, and collect insights for improvement of these systems.

The work concludes in part V (p. 335).

Part II

Fostering
human·collective·intelligence

Chapter II.1

Overview and structure

This part treats the first central concern of this book: fostering hucolligence (human-collective-intelligence). Before diving into the specific focus of this work in the overall challenge of hucolligence, it is useful to first investigate this overall challenge. This is done by investigating the meaning of hucolligence in chapter II.2 (p. 23). Subsequently, it describes and motivates the base-level system for fostering hucolligence, the Orcoba-Approach, in greater detail than has already been done so far (chapter II.3 (p. 43)). The Orcoba-Approach, however, is an abstract, base-level system design that cannot be effectively applied as such. For this purpose, it needs further specialisation ('extension'), which, depending on the context in which the Orcoba-Approach is applied, can be done in different ways (chapter II.4 (p. 49)). One of the essential extensions of the Orcoba-Approach consists of *evolutionary inspired extensions*. These provide an adequate answer to several major challenges, such as how to effectively deal with the limited human resources for composition-record development and assessment. Its sub-extensions, inspired by evolutionary biology and evolutionary computation, therefore, are treated elaborately (chapter II.5 (p. 65)).

Chapter II.2

Introduction

(Before starting this chapter, I would like to gently remind the reader of the call for feedback on page i.)

This chapter first elaborates on the meaning of the notion ‘hucolligence (human·collective-intelligence)’. Amongst other things, it elaborates on its ‘definition’, different dimensions of it and why it cannot be designed, but only can be fostered. Then it briefly treats the focus of this work within the process of fostering hucolligence.

II.2.1 What is human·collective-intelligence?

First a note to the reader: In the literature, the broader notion collective intelligence is used far more often than human·collective-intelligence. Therefore, in the references throughout this chapter, the literature covering the notion collective intelligence is tacitly included as well.

Hucolligence can, in my opinion, not be defined, but only be characterised. That is, one can describe an ever-extensible set of distinctive characteristics or essential features of hucolligence, without ever being capable of grasping the entire notion in words only. ‘Definitions’ of hucolligence, and the broader notion collective intelligence in the literature are, in my opinion, rather characterisations than definitions. Examples include: “the ability of a group to solve more problems than its individual members” (Heylighen, 1999). “groups of individuals acting collectively in ways that seem intelligent” (Malone and Bernstein, 2015).

In spite of this, I do not mean to say that hucolligence is an illusory notion. A brief philosophical reflection on semantics in general may elucidate my view. In general, a definition of a word or a word-sequence in a language consists of another

word or word-sequence the meaning of which is already assumed to be known by the reader. In their turn, these words are defined in terms of other word(-sequence)s, and so forth, until you reach word(-sequence)s the meaning of which was established by direct experience or exposure to their usage. In the end all word(sequence)s must reduce to the latter, so to direct experience. This is related to the so-called symbol grounding problem (Harnad, 1990). These word(sequence)s are in a sense ‘language atoms’ – they cannot be defined in terms of language. The notion compositionality of semantics also serves a role in this reflection. A word-sequence has a compositional semantics, if the semantics of the sequence is equal to a function of the syntactical structure of the sequence and the meaning of the individual words, and nothing more (Szabó, 2017; Partee, 2008). In other words, the semantics of the word-sequence can be reduced completely to the semantics of the individual words – the sequence can be analysed ‘reductionistically’.

These reflections imply that real ‘definitions’, in the sense of a complete and self-contained description of something, can never been given in natural language. However, in a certain sense, one get close to something that resembles a definition. This is the case if the ‘definition’ consists of one word with a strongly univocal and precise meaning to most speakers of the language, or a word-sequence that (1) is composed of words with a univocal and precise meaning and (2) that has a compositional semantics (*end of list*). An example is the definition “A circle with a dot in its centre”. The meaning of the words in the last sentence are relatively clear and univocal. The structure of the sentence will also be clear to most people, and the meaning of the sentence is, to the best of my judgement, compositional.

Unfortunately, I believe that the word-sequence human-collective-intelligence itself is neither compositional, nor a sequence the constituent words of which have a precise and univocal meaning. Moreover, I believe there is no alternative word-sequence with this property that can serve as its definition. First and perhaps foremost, the most essential word of the phrase: ‘intelligence’ does not have a precise and univocal definition. This can, among other things be derived from the numerous attempts to describe it. Psychometrist Robert Sternberg seemed to have stated: “there seem to be almost as many definitions of intelligence as there were experts asked to define it.” (Legg et al., 2007). Malone and Bernstein (2015) in their pursuit to define collective intelligence even abstain from committing to a particular definition of intelligence because “there are so many ways to define it”. My direct and philosophical argument is that it requires intelligence to be able to define anything. Hence, intelligence itself can not be reduced to a definition. In other words, intelligence is presupposed in the notion ‘definition’ itself, and trying to define it would – in essence – lead to a circular and therefore essentially void definition.

Nevertheless, I do not believe that the attempts to define intelligence are meaningless. I think that something remarkable happens in the process of these attempts.

I believe the following reflection to, in essence, hold for any language expression. Language is not so much a means to define things, but a system that is in continuous co-evolution with the world that surrounds it. In doing so, it is a means that reflects certain aspects of things in the world, more so than that it defines them. In other words, if one talks about a 'definition of intelligence', one is actually talking about a reflection of an aspect of intelligence into the systems of writing and speech. Reading a 'definition' of intelligence is, from this perspective, similar to looking into a mirror. A mirror neither defines nor comprises who you are by standing in front of it – it merely reflects an aspect of who you are. Even the image of the reflection is not 'in the mirror', but in the interaction between your perception and consciousness and the mirror. If you stop looking into the mirror, the reflection as meaningfully perceived by you also ceases to exist. Analogously, a sentence describing intelligence does not define intelligence, but merely reflects a part of what it is back into the intelligent observer itself as long as that observer is interacting with that sentence. One can recognise it as a 'definition' of intelligence because it resonates within one's mind with aspects of what one conceptualises as intelligence.

Reflection, however is not useless. Just as a good mirror can help you investigate yourself, a good reflection of aspects of intelligence in language can help investigate and foster intelligence. Another interesting analogy can be found in molecular biology and genetics, and particular in epigenetics (Holliday (1994) defines the latter). Traditional belief is that genes can be interpreted as a 'code book' or 'blue-print' that defines and describes the functioning and structure of a given organism, amongst which its phenotype. Hence, one believes in one-way traffic that leads from genes to phenotype with as intermediate steps, among other things RNA transcription and protein creation (Watson et al., 1953). However, many contemporary scientists harbour a radically different view – such as for example Davies (2012, starting at 00:04:20) who speaks of the 'blue-print fallacy'. They believe, and have shown, that the genes of an organism and its macroscopic structure and function are in a continuous mutual and circular interaction: it is a matter of two-way traffic (Parrington, 2017; Moore, 2015; Li et al., 2015). Both the macroscopic structure and form influence the gene expression through, among other things, epigenetics, and vice versa, genes influence the macroscopic structure. Some go even further by including into this macroscopic structure the environment and ecosystem in which the organism lives. Not one of them is more fundamental and can be reduced to the other or interpreted as the definition or design of the other aspect. I believe that the relation between a language and its context, the humans using it, is very similar. As said, I also believe that the phrase human-collective-intelligence does not have a compositional semantics. In other words, the meaning of the phrase has a meaning that cannot be exclusively deduced from the meanings of the individual words and the structure of the phrase. I believe that the meaning is at least partially rooted in direct exposure

to the phrase as a whole, and it cannot, in its entirety, be reduced to other words.

Therefore, the remainder of this chapter proposes a number of characterisations of hucolligence instead of an attempt to define it. It starts with characterisations of the word ‘intelligence’, and then the word ‘human’ and ‘collective’ in relation to intelligence.

II.2.1.1 *Intelligence*

Phrases that I believe to non-exhaustively characterise intelligence are as follows. Note, that in the list I do not take into account the ambiguous nature of the word intelligence.

- | | |
|-----------------|--|
| Reaching goals: | being able to reach one’s goal, solve problems by interacting with the environment. As Goertzel (2006) states: “achieving complex goals in complex environments”. |
| Non-dogmatism: | an open and non-dogmatic attitude in which one is willing to scrutinise one’s current understanding, beliefs and opinions and revise them in the light of new insights. A closely related statement made by Aristotle (0BCEa) is that “it is the mark of an educated mind to be able to entertain a thought without accepting it”. |
| Adaptability: | the capability to adapt one’s environment and to one’s environment to be able to cope more effectively with that environment. As Sternberg (2017) states, “psychologists have generally agreed that adaptation to the environment is the key to understanding both what intelligence is and what it does.”. |
| Consciousness: | intelligence and consciousness are closely related. True intelligence requires consciousness. For example, Chella and Manzotti (2007) state that consciousness “corresponds to many aspects of human cognition which are essential to our highest cognitive capabilities: autonomy, resilience, phenomenal experience, learning, attention”. |
| Generalisation: | being able to transfer lessons learnt in a certain situation to a different situation. |

Logical reasoning:	the process of reaching conclusions by applying valid inferencing rules to a given set of premises.
Multiplicity:	from a certain perspective, one can distinguish multiple forms of intelligence. For example, Gardner (2011) distinguishes 8 different types of (human) intelligence, amongst which bodily kinesthetic, mathematical-logical, intrapersonal and existential intelligence.
Multi-facetedness:	from a certain perspective, intelligence has a multi-faceted nature. Anastasi (1992) states that “intelligence is not a single, unitary ability, but rather a composite of several functions.”

As said, I did not yet take into account the ambiguities of the word ‘intelligence’. For example, in contrast to the multi-faceted nature of intelligence referred to in the aforementioned, one could also postulate that there is something that pervades, or lies at the origin of these multiple facets and call that the one true ‘non-faceted’ intelligence. This is somewhat related to the discovery of the *g* factor, which shows that there is a positive statistical correlation between the performance on a wide range of tasks that require intelligence (Spearman, 1904). This would suggest that there is a single, underlying form of ‘general intelligence’ that cannot be measured directly.

Another interesting angle to approach the definition of intelligence is by focusing on the elaborative instruments that have been developed to measure general intelligence in the field of psychometrics. See the work of Kline (2013) for an overview of some of the most important instruments that have been developed in this field. As Boring (1923) stated: intelligence is what intelligence tests measure. Even assuming that there are well-designed and scrutinised intelligence tests available, I do not agree with this. I believe that these tests suffer from the same limitations as any other attempt to ‘define’ intelligence, that already have been put forward in this section. However, in contrast to characterisations in natural language, by ever extending and improving the collection of well-designed intelligence tests we could perhaps come closer and closer to an objective instrument to reliably identify at the least some forms of intelligence, without ever reaching it.

II.2.1.2 *Human·collective·intelligence*

Suppose that we assume the meaning of the word ‘intelligence’ to be known, how does adding ‘collective’ modify its meaning? I believe that a good general characterisation is: collective intelligence occurs when the intelligence of the collective is greater than

the sum of the intelligence of the individuals. The difference between the first and the latter can be called the ‘collective’ aspect of the intelligence of that group. Another is that collective intelligence requires an intelligent interdependence between the intelligent behaviours of the individuals within that collective. The greater the degree of this interdependence, the greater the degree of ‘collectiveness’ in intelligence. For example, a book that is translated by assigning each chapter to another individual translator, is translated in a less ‘collectively intelligent’ way, than the same book being translated in a way in which all translators collaborate in translating each chapter of the book, for example by discussions and carrying out complementary tasks on the same piece of text, such as editing, reviewing and translation. As Heylighen (1999) states, a “group can be said to exhibit collective intelligence if it can find more or better solutions than the whole of all solutions that would be found by its members working individually”.

self
organisa-
tion

Other, specific features that are typically attributed to a greater degree of collective intelligence are decentralisation and the related term self organisation. In *self organisation*, a term coined by cybernetics pioneer Ashby (1947), complex global structures emerge from simpler local interactions of agents, without centralised control (Heylighen, 2013; Heylighen et al., 2001) *Example: The market economy can be considered as a form of collective intelligence the decentralised structure of which demonstrably contributes to certain qualities. Centralised planning of the economy would not be able to keep up with the calculations needed to reach an equilibrium, as a opposed to a decentralised approach (Gode and Sunder, 1993).* If one sees the degree of decentralisation and self organisation as one of the hallmarks of collective intelligence then, interestingly enough, one could assign social insects, such as bees and ants a higher level of collective intelligence than typical groups of collaborating humans. The difference between the ‘intelligence’ of each individual insect and the intelligence of the ‘hive’ seems at least to be greater than that of an human individual and a typical human team

II.2.1.3 Related research

A problem with the field of collective intelligence is that there are a multitude of disciplines and research groups that are devoted to research into it or closely related to it, without necessarily explicitly and frequently using the term. This section covers a (non-exhaustive) selection of important research disciplines and fields. It also provides examples of interesting findings related to collective intelligence for several of the mentioned disciplines and research groups. Note that these findings are not necessarily applied in the systems I developed in the scope of this study, although they provide inspiration for future extensions. Its purpose is, primarily, to provide an overview of the topic of fostering collective intelligence beyond the focus of this

book.

II.2.1.3.1 Collective intelligence-research

There is a limited number of researchers who have explicitly associated their work with collective intelligence, using exactly those words or synonyms thereof. Researchers in this field are, typically, most comprehensively involved in the question of how to foster collective intelligence in the sense that they try to address many facets of collective intelligence, whereas most fields that follow in the rest of this section typically only address certain aspects of it. These researchers are, therefore, often characterised by a much more interdisciplinary attitude, often combining insights from several fields. Nevertheless, different research groups still seem to emphasise certain disciplines more than others.

No survey about collective intelligence is complete without mentioning the work of Engelbart, who was a pioneer of contemporary collective intelligence research. He successfully employed a structure of a self-reflective organisation in which the goal of the group was to improve its own intelligence by fostering its own co-evolution of human capabilities and artefacts, enabling a positive avalanche effect (Engelbart, 1962). This structure formed the core principle of his work. His lab produced an astonishing number of spin-off products that were about 30 years ahead of his time. He and his team invented the probably first version of the World Wide Web (later rediscovered by Berners-Lee (1999)), and the modern graphical user interface. Unfortunately, the public awareness is limited to these spin-offs, in particular the invention of the computer-mouse. To Engelbart these were all of secondary importance, and, from his perspective, merely transient stages in fostering collective intelligence. He developed notions such as collective IQ (Engelbart, 1995). The work of Engelbart strongly leans towards the technological aspects of collective intelligence, and their complementary human capabilities.

Many of the basic principles of this book resonate with the work of Engelbart. In fact, in retrospect I had very similar ideas. During a literature study I discovered his work, and an immediate recognition followed. It further corroborated my view that the designs developed in this book can be fruitful when actually applied in society.

A contemporary, and possibly the world's largest institute that is devoted directly to research in collective intelligence is the MIT Center for Collective Intelligence. For example, Woolley et al. (2010) provided empirical support for the hypothesis that a form of 'general collective intelligence' exists for groups, for which they coined the term *c factor*. This is a generalisation of the earlier mentioned study of Spearman (1904) into the *g factor* that showed the same for individual persons. The *c factor* is an hypothesised general ability of a group to perform on a wide variety of tasks. Empirically, it is measured by determining the statistical correlation between the *c factor*

group's performance on a wide variety of tasks. The study was conducted among about 700 testpersons working in small groups. One interesting conclusion from the study was that the c factor was, perhaps surprisingly, not strongly related to the average or maximum intelligence of the group's members. The work of the MIT Center for Collective Intelligence leans strongly towards organisational studies and organisational psychology. Another institute is the Global Brain Institute (Heylighen, 2002) based in Belgium. They focus on fostering the development of what they have coined 'the Global Brain', a form of distributed intelligence emerging from the Internet. The institute uses scientific methods to "better understand the global evolution towards ever-stronger connectivity between people, software and machines in service of a course towards the best possible outcome for humanity".

II.2.1.3.2 Cybernetics

cybernetics

Cybernetics is the field that concerns itself with research into abstract theories of complex systems (Ashby, 1956; Wiener, 1961; Heylighen and Joslyn, 2001). It abstracts from the substrate of the system, and focuses on the form and the functioning of the system, whether it be a system in, among others, psychology, economy, artificial intelligence, sociology or biology. As Paul Pangaro (2018) points out, it is confusing that there is another interpretation of the word cybernetics in popular 'SciFi' contexts. A particular interpretation is that of biomechanics, such as chip implants into humans. Here, however, cybernetics is interpreted as the rigorous and abstract discipline founded by the mathematician Wiener. The basis of the model is formed by considering the way a given system uses information, models and control actions to steer towards, and maintain goals (Heylighen and Joslyn, 2001). Cybernetics itself has not established itself as a common discipline, such as for example computer science or sociology. For example, almost every university offers a study computer science, while there are few that offer cybernetics. Heylighen and Joslyn mention as the major reason the high level of abstraction and transdisciplinarity of cybernetics. As a consequence, more narrow and therefore more graspable specialisations and application areas, such as control engineering and subdisciplines within computer science have established themselves as separate disciplines, drawing away almost all human resources from cybernetics (Heylighen and Joslyn, 2001, Sec. I.C, p. 4). Nevertheless, indirectly it has exerted a great influence and its fruits have found their way into society, among other things by means of these specialisations. The relations with the Orcoba-Approach include the following. First, cybernetics provides indirect support for several essential design decisions of the Orcoba-Approach. Among other things, a fundamental principle of cybernetics is the presence of adequate feedback loops in a system, allowing it to self-regulate. In Orcobas this is related to the principle of institutionalised-self-reflection. The Orcoba-Approach, in a sense, facilitates and

encourages the highest form of feedback possible: that of conscious beings reflecting and steering their own process. This is closely related to *second-order cybernetics* within which the observer and modeller of the complex system is regarded as being part of the system (Heylighen and Joslyn, 2001, Sec. I.B). Among other things, the goals and associated control loops within a given system, and even the model of the system, are then adapted by the system itself. Second, it provides support for several design decisions in specialised extensions of the Orcoba-Approach. For example, one specialised extension is inspired by the theory of evolution and evolutionary computation (see chapter II.5 (p. 65)). The field of evolutionary computation is closely related to cybernetics, it defines a system that self-regulates towards a goal using a quantified feedback loop in terms of a so-called fitness function – probably the field of evolutionary computation has also been influenced by cybernetics. *second-order cybernetics*

II.2.1.3.3 Game theory

Formal (mathematical) models of groups of interacting people have long been used to gain insight in social phenomena, including those related to hucolligence. A tremendously influential sub-field that employs such models is game theory (Neumann, 1928). *Game theory* can be defined as “the study of mathematical models of conflict and cooperation between intelligent and rational decision makers” (Myerson, 2013). A classical example is the Nash equilibrium in game theory (Nash et al., 1950; Osborne et al., 2004), which has a profound influence on decision making within groups, and therefore their collective intelligence. Note that game theory is commonly regarded as a subdiscipline of economics. However, it can also be interpreted as a subdiscipline of applied mathematics with important applications, among others, in economy, biology, computer science, sociology and philosophy. *game theory*

Game theory is strongly related to collective intelligence. Among other things, it can form an instrument to help determine which regulations are most likely to promote collective intelligence of a group. For example, Wood (2011) demonstrated by means of game theory that certain regulations would lead to strategic situations that would promote cooperation in reducing global greenhouse gas emissions, while others would not. Implementing the first type of regulations, hence, would lead to a greater degree of collectively beneficial behaviour, which could be regarded as a greater level of collective intelligence.

II.2.1.3.4 Social simulation

Social simulation is the discipline that “uses computer simulation in the social sciences” (Gilbert and Troitzsch, 2005). In other words, it develops and applies computation-based simulations of social groups, such as human groups or societies. The purpose *social simulation*

is, among other things, to increase our understanding of social phenomena occurring in such groups or to steer or influence such phenomena. The field is interdisciplinary, both with regard to the development of the simulations and the phenomena it addresses. In the development of simulations, social simulation draws from a wide range of disciplines including computer science, sociology, psychology, economy, philosophy, physics, engineering and mathematics. It addresses phenomena occurring in, among others, sociology, economics, and anthropology (Edmonds and Meyer, 2017; Wikipedia, 2018d).

*micro-macro
link*

Typically, social simulation is based on the principle of emergence in complex systems: it provides an instrument to investigate how microscopic behaviours, typically the behaviours of the individuals within the group, lead to macroscopic phenomena (Squazzoni, 2008; Edmonds and Meyer, 2017; Wikipedia, 2018d). In other words, it establishes the so-called *micro-macro link* in sociology (Alexander, 1987). In social simulation it are commonly these microscopic behaviours that are represented in a computer model. The simulation consists of ‘running’ the model and allows one to induce macroscopic phenomena from the model. Typically, the microscopic behaviours that are modelled are those of the individuals of the group. In other words, the model is very fine-grained.

For example, social simulation has refuted the once popular hypothesis that selfish individuals will always be favoured through natural selection (Axelrod, 1997b). The field overlaps with so-called multi-agent systems (Wooldridge, 2009) (see section II.2.1.3.5 (p. 34)), which typically also establishes a micro-macro link. Moreover, it closely resonates with the field of cybernetics, because it typically focuses on modelling phenomena related to decentralisation, open systems, self-organisation, and emergence. Social simulation has a number of advantages over other fields that are also based on formal models to gain insight in social phenomena, such as game theory. The other fields are mainly based on the so-called ‘analytical’ approach. In the analytical approach, a formal model is created and used to mathematically deduce phenomena occurring in the group (Axelrod, 1997a). Social simulation is closely related, however, instead, its formal model is represented in the computer as a basis for a computer-simulation of the group. From the result of the simulation one then attempts to induce, and not deduce, phenomena occurring in the group. The great advantage of simulation over deduction is that it is computationally far less resource intensive. Creating deductions from microscopic behaviours ‘all the way’ up to the macroscopic level is in most cases infeasible: the computational gap is simply too large. In the analytical approach one has to resort, therefore, to creating a formal model on a more macroscopic level. This requires assumptions that are more tentative, because macroscopic levels can often not be observed directly, in contrast to the microscopic behaviours. Social simulation allows one to close the micro-macro gap (Edmonds and Meyer, 2015, p. 4).

At first sight, the Orcoba-Approach and social simulation or mathematical modelling of human groups seem to be disjoint. After all, an Orcoba is a real human group, and not a mathematical or computational model of such a group. Nevertheless, social simulation could be applied in several ways. First, certain design decisions within a composition-record can be explored and fine-tuned with such models. This allows one to investigate the effect of these design decisions before actually adopting them in a composition-record and expose them to human groups. This may save human resources and speed up the evolution of Orcobas. Typically, a social simulation could provide support for which micro-level behaviours being made explicit in a composition-record lead to desired emergent macroscopic effects. *Example: In one of the specialised extensions of the Orcoba-Approach, SWiFT-Fixtal (chapter III.7 (p. 225)), participants have to collaboratively translate a text into a deductive·formal·language. In this process, they have to judge the language atoms they want to introduce into the language (the ‘nodes’) and vote them in or out. Ideally, each participant judges all nodes so that unanimous agreement can be reached. However, obviously there is a trade-off between judging nodes and translating texts. If the text and the group of participants is very large, then it will even be impossible to demand that each participant judges all nodes: they would never come to actual translation work. In this case, social simulation could help to find out how to strike a good balance between the two activities, and optimise the effort spent on voting. In contrast, an ‘in vivo’ approach with real human groups would be quite human intensive.*

Second, lessons learnt from other social simulation experiments and mathematical models could provide indirect support for design decisions. This application of social simulation is closely related to what Edmonds et al. (2017) call ‘illustration of ideas’ – the simulation does not so much prove that a phenomenon holds for a given real group, but shows that, under certain assumptions made in a model, a phenomenon will occur. It is also related to the way Hales (2013) attempts to apply insights from social simulation to formulate design principles for effective distributed computer systems – the application area, distributed computing is different, but the approach is the same. An example of a social simulation that could provide indirect support for certain design decisions is the work of Gode and Sunder (1993) that demonstrates that there are economical models in which even if individuals exhibit simple behaviours, the collective still converges towards a global optimum. In other words, in this case the emergent intelligence of the crowd is in a sense, much larger than the intelligence of the individuals that make up this crowd. Another class of examples is how to effectively deal with resource limitations. For example, Singh and Haahr (2006); Hales (2013) describe how to apply the social segregation models of Schelling (1971) to come to an efficient network topology in peer-to-peer networks of computers. Such a system could also be applied within certain Orcobas. Examples are the Orcobas developed in this book to foster formal·fluency such as SWiFT-Fixtal (see chapter III.7

(p. 225)). As stated earlier in this section, a problem that may occur in these Orcobas is that if the body of knowledge being expressed in a formal-language becomes large, it is hard to prevent double occurrences of nodes. After all, a given translator may not be aware that there is already a node with a given meaning, and may introduce another node with the same meaning. This leads to fragmentation of the KR-base, and reduces the efficiency of automated-deduction. A self-organising set of rules such as applied by Singh and Haahr (2006) could be applied to effectively minimise the occurrence of such double nodes by self-organising an efficient peer-reviewing scheme, which determines which translator reviews what node-definitions.

II.2.1.3.5 Multi-agent system and normative multi-agent system

multi-agent system A *multi-agent system* (MAS) is a group of artificial agents that interact with each other in order to achieve goals. These goals can range from goals pursued by individual agents to common goals, and can be mutually conflicting. To achieve their goals, the agents must be able to effectively cooperate, negotiate and coordinate (Wooldridge, 2009). One relation between MASs and Orcobas has already been made clear in the previous: social simulation. Another one is with a specialisation of MASs: normative multi-agent systems, the explanation of which follows now.

normative multi-agent system *Normative multi-agent systems* are formed by the intersection of normative systems (Raz, 1999) and multi-agent systems (Boella et al., 2006). Note that it, therefore, can also be regarded as a specialised form of social simulation – one that restricts its focus to behaviours and phenomena in relation to norms. An interpretation of the word ‘norm’ is that it is a principle of right action binding upon the members of a group and serving to guide, control, or regulate proper and acceptable behaviour (Boella et al., 2006). Interest in such systems has risen in recent years, among other things due to the fact that there is an increasing need for computer systems that exhibit some form of automated ‘normative and ethical’ reasoning, such as self-driving cars, automated trading systems and high-frequency trading (Davis et al., 2013), and expert systems used in medicine (Collste, 1992). For example, if a pedestrian suddenly crosses the street, a self-driving car may be forced to make an ethically loaded decision: to either protect the driver and injure the pedestrian by continuing driving, or protect the pedestrian and injure the driver by breaking violently.

The field seems not yet to have converged to, and probably will never converge to a standard overall definition. This is in part due to the ambiguous and context-dependent nature of the word ‘norm’ and the difficulty to pin it down (Balke et al., 2013). Aware of this situation, Balke et al. provide several complementary definitions. The most concrete and technical one is that “a normative multi-agent system is a MAS organised by means of mechanisms to represent, communicate, distribute, detect, create, modify, and enforce norms, and mechanisms to deliberate about norms

and deter norm violation and fulfillment” (Balke et al., 2013, Sec. 3.3).

Typical examples of norms are that one should not steal from others, that one should not bump into other people when walking on the street.

Relations between the Orcoba-Approach and normative multi-agent systems follow now. First, as Boella et al. (2006) state, the use of norms is a key element of social human intelligence – in other words, hucolligence. Second, the capability-constitution of an Orcoba can be considered to be a normative system. Therefore, a normative multi-agent system can be considered as a computational model of important aspects of an Orcoba. Insights from the study of normative multi-agent systems could, therefore, be applied to Orcobas. For example, Grossi et al. (2010) discriminate between norm-creation and norm-implementation: simply prescribing the norms agents ought to follow (norm-creation), does not imply that the agents will indeed follow these norms (norm-implementation). Norm-implementation can be realised by dictating the norms (“norm regimentation”), i.e. making sure that agents cannot deviate from them, or by a more mild approach, in which agents are encouraged to follow them to a stronger or lesser degree, for example by varying degrees of law enforcement. The second is often favourable, because the system will then become more flexible and therefore more collectively intelligent. For example, it is not always good to follow all prescribed norms, and agents should have the freedom to deviate from them depending on the circumstances. Grossi et al. (2010) investigate formal models to gain insight in these forms of norm-implementation. If one considers a capability-constitution as a result of norm-creation, these types of insights could possibly be applied to implement these norms more effectively.

II.2.1.3.6 Organisational studies and educational science

Several specialised disciplines within the field of social sciences are related to collective intelligence research, amongst which the general areas sociology and social psychology. In particular, this is the case for the following subfields: organisational studies, organisational psychology and educational science. The subsequent elaborates on organisational psychology.

Organisational psychology is “the science of psychology applied to organisations and the individuals working within these organisations” (Kozłowski, 2012; Bauer et al., 2015). A part of organisational psychology occupies itself with group dynamics, including factors that influence team performance, such as team effectiveness (Salas et al., 2009) and team composition – all closely related to collective intelligence. Examples of research results have already been introduced in section II.2.1.3.1 (p. 29), such as the work of Woolley et al. (2010) on the *c* factor. Other valuable research results related to team effectiveness are summarised by Salas et al. (2009), one of which is about shared leadership. Shared leadership is a kind of distributed leadership,

in which the leadership roles are changed dynamically by the team based on current challenges on the one hand, and the capabilities and traits of the members on the other (Yukl, 2013). It is to be contrasted with traditional leadership: vertical leadership. In an empirical study conducted by Pearce and Sims Jr (2002), teams with shared leadership were, on average, more effective than teams with vertical leadership.

II.2.1.3.7 Computer-Supported Cooperative Work and Computer-Supported Collaborative Learning

Computer-Supported Cooperative Work “is a generic term, which combines the understanding of the way people work in groups with the enabling technologies of computer networking, and associated hardware, software, services and techniques” (Wilson, 1991). An example of a valuable finding in this field is that so-called physiological synchrony in a team working together online, is predictive for the collective intelligence of that team. The collective intelligence here is operationally measured in the same way as done by Woolley et al. (2010), mentioned earlier in this section. Physiological synchrony is the co-occurrence in time of physiological activity among interacting individuals (Mønster et al., 2016). Examples are synchrony in heart rate, skin conductance and facial expressions such as smiling. It is measured by means of sensors and computers. Thus, this study produced results that suggest two ways to use computers to enhance collective intelligence: it provides a method of how to measure the effectiveness of computer-mediated collaboration, and therefore a way to improve this type of collaboration, and secondly, the method itself is also based on computer-technology.

Computer-Supported Collaborative Learning

A related notion is *Computer-Supported Collaborative Learning*, a definition of which can be obtained by replacing the word ‘work’ in the first sentence of the previous paragraph with ‘learn’. Stahl et al. (2006), for example, describe it as “a branch of learning sciences concerned with studying how people can learn together with the help of computers”.

II.2.1.3.8 The wisdom of the crowd

The notion ‘wisdom of the crowd’ seems to be used somewhat ambiguously. One usage is in the sense of the collective opinion of a group of people based on a statistical analysis applied to the aggregate of the independent opinions of each person in the group in those cases where the collective opinion can qualify as being superior to the opinion of a few. It is important to emphasise the word ‘independent’ here: one assumes there is no direct deliberation among individuals to come to their opinion. This is to prevent mindless copying of opinions between members of the collective, which can have an opposite effect: increasing collective ‘stupidity’. Surowiecki (2005)

popularised the term in this sense. An example he often mentions is the observation of Galton (1907) that the median of a large number of independent guesses of the weight of an ox during a game, was very close to the real weight of the animal. Retrospective analysis even revealed that the mean was exactly correct.

However, the term also seems to be used for any work done by a group of people where members of a large crowd each incrementally contribute to the creation of a single result, such as a Wikipedia-article, whether it be with or without intermediate deliberation and coordination. In Wikipedia there is clearly an intermediate deliberation among members of the crowd, in the form of discussion pages, but also by the mere fact that each contributor adapts the article that has already been written so far and therefore is exposed to the work of others before deciding what and how to contribute. Aristotle (0BCEb) provides the to my knowledge oldest treatise of the notion ‘wisdom of the crowd’, stating that “The many, of whom none is individually an excellent man, nevertheless can, when joined together, be better than those [excellent few], not as individuals but all together.” According to Ober (2013), Aristotle advocates the combination of deliberation and aggregation of independent opinions to create a wise crowd. Hence, he also clearly included deliberation in his interpretation of the notion wisdom of the crowd, in contrast to Surowiecki (2005).

Wisdom of the crowd is in popular terms often sloppily described as that the opinion of a crowd is better than that of individual experts. However, the literature displays a much more nuanced view. Several sources seem to agree that only for a specific type of problems and only when certain conditions are met, the crowd has a higher probability to outperform individual experts. For example, Aristotle (0BCEb) provides a detailed set of conditions that promote the degree of wisdom of crowds, amongst which that the group should be sufficiently diverse so that the expertise of the individuals complement each other sufficiently. Surowiecki (2005) presents four criteria that should be met, amongst which diversity and decentralisation: each member should be able to draw from local knowledge. Oinas-Kukkonen (2008) postulates that communication should be limited to a certain degree, however, that on the other hand contest and disagreement should be encouraged.

II.2.1.3.9 Swarm intelligence

According to Kennedy (2006), swarm intelligence refers to a “problem-solving ability that emerges in the interactions of simple agents”. The swarm’s problem-solving ability goes beyond that of the individual agents. The agents can be artificial (robots, computer-programs) or biological (insects, birds or human beings). The notion was introduced by Beni and Wang (1989). The term is mostly used in the context of fostering collective intelligence, and not in descriptive studies into, for example, already existing biological swarms, such as flocks of birds. The latter are often used

as inspirations for the design of swarms. The field seems closely related to the field of multi-agent systems, when assuming the broader interpretation of a MAS as one that includes non-artificial agents. At first sight there do not seem to be many differences, except for each having originated from a different community of researchers. However, on closer inspection one could argue there are some notable differences. First, the goals of both research fields are different. The presence of the word ‘intelligence’ in swarm intelligence indicates that it focuses on fostering collective intelligence. In a sense, MAS-research has a wider scope: it includes any phenomenon of interest that emerges from the interactions between multiple agents, even if these would not contribute to a greater collective intelligence, such as some forms of conflict. In another sense it is narrower: MAS-research emphasises agents that have individual beliefs, desires and intentions (Georgeff et al., 1998). These agents try to fulfill their desires and intentions through cooperation or despite conflicts with other agents. In swarm intelligence, individual agents do not necessarily have individual desires or intentions. The focus is solely on a desired emergent collective phenomenon. In a sense, in swarms, agents are purely instrumental parts in achieving the collective goal, while in MAS they represent, to a lesser or stronger degree, individuals that also have and value an individual existence, possibly partially conflicting with that of others within the group.

In the interpretation of the wisdom of the crowd in which the intermediate interaction between members of the crowd is mimimised Surowiecki (2005), wisdom of the crowd and swarm intelligence also differ from each other (Rosenberg et al., 2016). In swarm intelligence there is a mutual interaction and coordination between agents while they are working towards a common goal, while in wisdom of the crowd this interaction is absent, and one suffices with applying a statistical analysis to the aggregate of decisions made by each agent individually. Note, however, that there are also other interpretations of the notion wisdom of the crowd (see section II.2.1.3.8 (p. 36)).

human swarm- A part of the field explicitly dedicates itself to swarms that contain human agents: *ing human swarming* (Rosenberg, 2015). This subfield is directly linked to the core topic of this book: fostering human-collective-intelligence. A result from this field is that the investigated swarms outperformed systems that do not employ mutual interaction and coordination while agents were working toward a common goal (Rosenberg et al., 2016).

II.2.1.3.10 Some other research fields

The survey provided in this section does not do justice to all fields that are strongly related to collective intelligence. Other fields include, among others: social choice theory, which is the study of collective decision processes and procedures.

II.2.1.4 Dimensions

Hucolligence is not one monolithic entity: when it manifests itself, one can distinguish several mutually interacting dimensions within it, each of which may be present in it to a lesser or stronger degree. In this subsection, I focus on the dimensions that are particular to *collective* intelligence, so not on the dimensions of intelligence as it is in itself, which self-evidently are also a part of hucolligence. ‘Top-level’ dimensions of hucolligence include, but are not limited to the dimension of human capabilities, the technological dimension and the transcendental dimension.

II.2.1.4.1 The dimension of human capabilities

The dimension of human capabilities can be further subdivided into, among other things: the cognitive, social and the contextual subdimension.

The human cognitive dimension relates to the role of human cognition in hucolligence. ‘Cognition’ is another word that is vaguely defined, but for the purpose of this work, the following characterisation is sufficient: it relates to the human capabilities that are at the heart of gaining knowledge and insight. It includes the invisible mental capabilities, such as language capability, memory, processing of sensory input, reasoning and intuition. Etymologically, the word *cognition* comes from the Latin verb *cognosco* (*con* ‘with’ + *gnōscō* ‘know’), itself a cognate of the Ancient Greek verb *gnōsko* meaning ‘I know’ (noun: *gnōsis* = knowledge), so broadly, ‘to conceptualize’ or ‘to recognize’.

The social dimension pertains to the degree of intelligence in the social interactions between individuals of a group. For example, if there is a low level of respect among individuals within a group, this is likely to negatively influence the hucolligence of that group. One can imagine that a lack of respect will also influence at least some other dimensions negatively, for example the cognitive dimension may suffer under it. Sub-dimensions of the social dimension include (aspects of) the economical and political dimension.

The contextual dimension refers to the ability to place one’s activities in broader perspectives, and conduct in such a way that, seen from these perspectives, one’s activities can also be considered to be wise. *Examples: (1) A human collective that is highly capable of building fossil fuel burning cars may be highly intelligent along the cognitive dimension. However, the group scores very low along the contextual dimension when it is not able to put a halt to this process at the moment it discovers this process disturbs the climate to such an extent, that it is detrimental to their own survival. (2) An example on a more individual level is: not eating sufficiently may save time in the short run, but in time one will weaken and lose time.*

II.2.1.4.2 The technological dimension

The technological dimension of hucolligence exists in two senses: broad and narrow. The broad sense refers to artefacts in general such as human writing, drawings, clothes and human glasses. The narrow dimension limits this to artefacts that consist of highly sophisticated technology as we know nowadays (the computer, but also the bicycle, piano etc.). The technological dimension – in its broad sense – is an intrinsic part of (collective) intelligence, that is to say, there is in my opinion no collective intelligence possible without that dimension. Even bacteria manipulate chemicals in their surrounding environment to ‘build artefacts’. In its narrow sense, the technological dimension has gone through a tremendous growth in the last century with, among other things, the invention of the modern computer. The technological dimension, and in particular networked information technology, therefore, is at this point in history an important point of focus with regard to fostering hucolligence. If one wants to maximise the effect of research and development efforts in hucolligence, the technological dimension is an obvious choice. This is, therefore, part of the focus of this work.

II.2.1.5 The transcendental dimension

The transcendental dimension refers to my idea that hucolligence exists by virtue of a greater collective intelligence in which it is embedded. Ultimately, the hucolligence of a group of humans, or for that matter for any collective or individual intelligence, is not a separate, isolated thing. It is inextricably intertwined with the larger systems of which it is part: human society, Earth’s ecosystem, the Earth, and in its turn the Earth from its place and interactions with the universe. In my opinion, attributing intelligence to a single unit, whether it be an individual human being or a collective is incorrect from a fundamental perspective. Related to this are statements such “intelligence is not a property of systems, but a property *between* systems.” (Paul Pangaro, 2018). This also works downward: each intelligence consists of smaller ‘intelligent’ interacting units, and so, all intelligence is in a sense ‘collective’. This structure is, for example, plausible for human intelligence. From a certain simplified perspective, each human being can be regarded as many billions of collaborating cells.¹

¹Of course, it is an open question whether intelligence can arise from smaller, non-intelligent units, as some kind of emergent property. In the particular case of human cells, and in particular neurons, there is disagreement among neuroscientists whether these cells are non-intelligent – or the opposite. A second problem is whether one arrives at the smallest conceivable intelligent ‘atoms’ at a certain point. I think these atoms do not exist – not so much that there is an infinite regression, but that it is more the perspective that creates the illusion of an infinite regression, or the necessity for introducing atoms. In the end, even these atoms deduce their meaning from the context they are part of – and perhaps, from

II.2.1.6 Fostering hucolligence

The central goal of this research project is fostering hucolligence. Before explaining that in detail, I first want to elaborate on the question why I call it ‘fostering hucolligence’, and not ‘creating or designing hucolligence’. According to the dictionary ‘to foster’ means “encourage the development of (something, especially something desirable).” First, hucolligence exists as long as humanity has existed and, in the form of (non-human) collective intelligence, even beyond that. You cannot design or construct it, but rather aim to stimulate the prosperous growth of the existing forms.

II.2.1.7 Fostering co-evolution

As said before, in the whole spectrum of hucolligence, this work focuses on fostering the co-evolution of human capabilities and technology (the latter ‘in the narrow sense’). A leading insight in my approach for fostering growth of hucolligence is as follows. Technology can be regarded as an extension of the human body, and is, in a sense even indistinguishable from it. These new ‘body-extensions’ require co-evolution with other parts of the extended body, to fully exploit the new body part. (Note that next to our ‘normal’ bodies and technology, I consider the human cognitive processes as part of the extended body.)

Another way to phrase it, without the use of the notion ‘body extension’, is the following. From the perspective of this work, every new type of technology is only one dimension of a new human-technological system. This human-technological system, however, is still in its infancy when the first implementations of the new type of technology occur. In most cases it then takes time, and sometimes much time, before the co-evolution of the new type of technology and the complementary human dimension realises its full potential. I will call this process ‘reaching the fixed point² of the co-evolution of a certain new technology’. The reason for using this term is that it is a fixed point of the transformation process in which the new type of technology, and the human dimension influence each others’ development mutually (i.e. co-evolve). Note that this process can also be rephrased in terms of cybernetics (Heylighen and Joslyn, 2001, Sec. III.B): I believe that each new type of technology and its complementary human dimension can be considered as a dynamical system that moves towards a so-called *attractor* and ends in an *equilibrium* state.

As stated before this work focuses on hucolligence on two levels: (1) by developing the framework Orcoba and (2), by fostering formal-fluency by applying Orcoba to

attractor
equilibrium

another perspective, these atoms are the greatest units, so, containers, instead of constituents. The latter may, perhaps, give rise to a cyclic regression rather than an infinite one.

²In mathematics, a fixed point of a function is an element of the function’s domain that is mapped to itself by the function. In this case, it is used in the sense of a transformation that is repeatedly applied to its outcome until the outcome reaches a stable level, i.e. a ‘fixed point’.

formal-fluency. This second project can be seen in the light of fostering the mentioned co-evolution: it tries to accelerate the co-evolution of processes that have already started, towards their full potential. In this case, it focuses on the co-evolution of human language and reasoning with information technology. As I have stated. I deem it likely that we will evolve completely new ways of processing language and representations. However, it is often not foreseeable what the final stage will be and how to reach it. Therefore, this work does not purport to make all steps in the right direction. The actual trajectory, in retrospect, may differ from the vision developed in this work. Take, for example, Vannevar Bush's Memex machine. The Web came, but in a different medium than he had foreseen. Nevertheless, his thinking, and that of others may have been essential to the occurrence of networked computer technology as we know it nowadays.

II.2.2 Research questions

This book part focuses on investigating one of the overall research questions of this book, as defined in section I.1.3 (p. 7), and repeated below:

“

Research question 1 (the general quest). *How to foster human·collective-intelligence (or briefly: hucolligence)?*

(Quoted from page 7.)

”

Chapter II.3

The Orcoba-Approach

Orcobas were already briefly described in section I.1.4.1.2 (p. 9). This section describes and motivates them in greater detail.

For several reasons it is useful to consider Orcobas as a subclass of more general classes of systems. To wit: classes that also encompass systems that are less ‘open’ with regard to their composition-record. Reasons for this include: (1) in the process of applying the Orcoba-Approach to a certain capability, it may be efficient to use these more general systems before specialising and (2) explanatory purposes (*end of list*). These more general classes of systems are defined in this chapter. The following first defines the most general class, the Cobas, and then proceeds with defining relevant subclasses.

Definition 1 (Coba). *A Composition-Record Based Community (Coba) is defined as follows. A Coba is a community-pursuing-a-capability, together with a record that allows the capability-composition of their pursued-capability to be (at least partially) reproduced and examined. The term composition-record is coined for the mentioned record. The members of the Coba explicitly and exclusively use and follow their composition-record to practice the capability. They are the composition-record-users of the given composition-record.¹ Ideally, the composition-record contains parts pertaining to artefacts as well as human elements.* ■

Definition 2 (Coba-Approach). *The Coba-Approach is an approach which applies Cobas to foster human capabilities in the following way. In the approach, people (and other actors) actively seek to improve composition-records by changing them and observing the effect on the performance of Cobas. The term composition-record-developer is*

¹In the context of Game-Based-Orcobas, which will be treated later, composition-record-users are also called composition-record-players.

coined for these actors. ■

The word ‘record’ in ‘composition-record’ has been chosen for the following reason. ‘to record’ means: “To set down for preservation in writing or other permanent form.”, and ‘the record’ “the information or knowledge preserved in writing or the like.”² – exactly what is intended here.

The subsequent sections focus on several aspects of the Coba-Approach.

II.3.1 The human and artefact part

A composition-record ideally includes two complementary parts: a human part and an artefact part. They respectively pertain to the human capabilities and the artefacts playing a role in the capability of the community pursuing a capability. This is in agreement with the general principle of human-artefact-co-evolution of this work: the composition-record integrates the technological and the human dimension in one whole instead of focusing on one dimension only, allowing these dimensions to co-evolve.

II.3.2 Constitutional aspects and contributive aspects

Within a composition-record one can distinguish, orthogonally to the human-artefact dimension, constitutional and contributive aspects.

*constitutional-
aspect
capability-
constitution*

Constitutional aspects of a composition-record are a direct expression of (aspects of) the capability-composition. I coin the term *capability-constitution* for the whole of all these aspects (constitution for short). Note that the word ‘constitution’ in capability-constitution is used differently than in politics in a few ways, including the following. A political constitution only expresses human aspects, and not technological aspects. For example, a capability-constitution can contain a tool to use to solve a problem. A part of the capability-constitution, therefore, could be a physical hammer (or a specification of that hammer). A political constitution contains rules and laws to follow, while a capability-constitution, additionally, typically contains other descriptions, such as protocols or approaches helping one to solve problems. An example is a recipe of how to cook Sukama Wiki.

*contributive-
aspect*

Contributive aspects are aspects that contribute directly to a certain capability-composition coming into, or remaining in existence, while they do not form a direct

²<http://www.dictionary.com/browse/record>

expression of that capability-composition. I coin the term *contributive-capability-record* for the whole of all these aspects. An example is a set of exercises to practice playing golf. These exercises foster the human capability to play golf. The written account of these exercises, however, does not necessarily directly describe a part of the process of playing golf. Therefore, this account is not part of a constitution for playing golf. Another example is a training for musicians in how to follow the directions of the conductor of an orchestra. This fosters the musician's capability to do so during actual performances. The training as such, however, is not part of the capability-composition of giving concert performances, but it contributes to the creation of that capability-composition. An *analogy* is the composition of a delicious pie. This pie has been in the oven to prepare it. Fortunately, however, the oven is not an ingredient of that pie, as this probably leads to digestive problems. Rather, the oven has contributed to the creation of the pie's composition.

*contributive-
capability-
record*

Note that parts of the composition-record can exhibit both contributive and constitutional-aspect at once. For example, every constitutional text-fragment also has a contributive aspect, because it expresses or formulates aspects of the capability-composition in a *specific* way. The latter influences, among other things, the easiness with which different readers will be able to understand that text-fragment. This is also the reason I speak of constitutional and contributive 'aspects' rather than 'parts'.

II.3.3 The human part of a capability-constitution

The human part of a capability-constitution can be further subdivided into several elements, including: (1) procedures and protocols (2) definitions of a shared language, and (3) conditions and rules. These types can interact and overlap. Examples and elaborations are: (1) Procedures: an example of a procedure is a culinary recipe. (2) Definition of a shared language: an example is the word 'Fore' in golf, which is used to warn people that a ball has accidentally been hit in their direction. The shared language includes all forms of expressions that can be used to communicate, not only textual language. For example: shooting a gun means start running in sprint competitions; the man holding up his hand towards his dance partner, means grab it in salsa dancing. (3) Conditions and rules: This is about conditions to be met and rules to be followed by the people that are part of the community that are realisable by certain modes of conduct (e.g. always wear a red hat), or conditions concerning the selection of human individuals that make up the community (e.g. "50% must be women, 50% must be men" and "All participants must have a diploma from the vegetarian cooking academy 'Vlam In De Pan'").

II.3.4 The artefact part of a capability-constitution

The artefact part of a capability-constitution consists of the artefacts or a specification of these artefacts. It may be obvious to the reader that artefacts are often parts of a capability-composition. However, in some cases, they can also be part of a capability-constitution. This is because many artefacts meet the conditions of parts of a composition-record: it is possible to reproduce and examine these artefacts ('reverse engineer' them). Examples are forks and knives. Of course, this is not always the case. A non-example would be a computer, which is very hard to reverse-engineer. In that case, a specification is necessary.

II.3.5 Examinability of composition-records

A perfectly examinable composition-record is considered to be the ideal in the context of this work, because such a composition-record allows the inner workings of a capability-composition to be examined. This allows an active reflection and revision of it. This is in agreement with this work's general principle of institutionalised-self-reflection and decentralised-development.

*explanation-
degree*

The *explanation-degree* of a part of a composition-record is defined as the extent to which that part explains (a part of) the inner workings of the associated capability-constitution.

For example, the condition "The baking team should include a trained cook." has an extremely low explanation-degree: it is not clear how the trained cook 'works'. On the other hand, the condition "The baking team should apply the following recipes: etc. etc.", has a high explanation-degree.

II.3.6 Openness of Cobas

The Cobra definition does not yet specify the parties involved in creating the composition-records. According to one of the general principles of this work, decentralised-development, the participants of the Cobra should be involved as much as possible. Therefore, an extension of the Cobra for which the latter is the case is introduced in this section.

*Internally-
Open-Coba*

An *Internally-Open-Coba* is a Cobra in which the composition-record comes into existence in a socially decentralised process within the Cobra, hence, each member is either composition-record-developer or supporter of each part of the composition-record. This is directly in agreement with the general principle of decentralised-development. Moreover, an *Externally-Open-Coba* is a Cobra of which the composition-

record is open to the public for reuse. Motivation is as follows. This allows reuse by other people, so that they can benefit from it, adapt it to suit their own purposes (diversification) and participate in its improvement (evolution).

*Externally-
Open·Coba*

Finally, an *Orcoba* is a *Coba* which is both an *Externally·Open·Coba* and an *Internally·Open·Coba*. In other words, its composition-record is open for the public for reuse, and it is developed or supported by all members of the *Orcoba*. *Orcoba* is an acronym for Open Composition-Record Based Community.

Orcoba

II.3.7 Application areas

The *Orcoba*-Approach is a general approach to foster hucolligence. It is intended as a tool to foster a wide range of capabilities, however, it (and its extensions) will not be equally effective for all capabilities. A number of conditions have to be met. Most importantly, it must be possible to adequately express the capability in a composition-record.

Moreover, the *Orcoba*-Approach is an abstract top-level design, and typically only becomes really effective in its extensions. Each of these extensions may require additional conditions to be met, limiting the application scope per extension. For example, an important extension in this book are *Challenge-Based·Orcoba*, which can be most effectively applied if it is possible to create, on demand, good challenges to test the capability. When deemed important, these conditions are explicitly mentioned in the corresponding sections.

II.3.8 Research question refinement

The argumentation so far, among other things, provides a partial answer to research question 1 (p. 7): it is intended to make it plausible that the *Orcoba*-Approach is a promising general approach to foster hucolligence. Of course, it does not assume it is the only way, nor that it is an approach that is suitable to foster all forms of hucolligence. However, given limitations in (human) resources to carry out this study, this work limits its attention to the *Orcoba*-Approach.

The *Orcoba*-Approach, developed so far, is not only a *partial* answer in the sense just given, it is also an *unfinished* answer. The basic points of departure for fostering hucolligence were expressed by three principles: decentralised·development institutionalised·self-reflection and human-artefact-co-evolution, each based on the author's previous professional exposure, the literature and the work of people successful in fostering hucolligence such as Douglas Engelbart (see chapter I.1 (p. 3)). These principles, however, are very abstract, and it is not self-evident how to put

them into practice. The Orcoba-Approach satisfies these principles and makes them somewhat more concrete. It is, however, almost as abstract. There is still a long distance to cross from the ‘what’ to the ‘how’. Creating a system that actually *realises* these principles and the overall goal to contribute to increased hucolligence, gives rise to several subchallenges which are not trivial to solve. Among other things, some of the design ingredients of the Orcoba-Approach have to be made more concrete and operational, which leads to new extensions and variants of the Orcoba-Approach. This yields the following refinement of research question 1 (p. 7):

Research question 1.1. [*Hucolligence with Orcobas*] *What are extensions and variants of the Orcoba-Approach that significantly contribute to fostering hucolligence?*
[*Conclusion(s): page 126*] ■

The subsequent chapters of this part focus on this research question.

Chapter II.4

Orcoba·extensions and generalisations

The Orcoba-Approach does not describe a single concrete system, but a class-of-systems: “the” Orcoba-Approach is a description of the most essential design decisions all systems that belong to this class share in an abstract way. The (different) specialised extensions are intended to make the general abstract notions concrete in possibly different ways. This study applies this notion of classes of systems for two reasons. First, it allows the presentation of a concrete, detailed, system in a layered, and more understandable, way. Second, different applications may require different specialisations. For this reason, this book presents several extensions and generalisations of the Orcoba-Approach, an overview of which is presented in chapter II.4. Therefore, from now on, this book uses the description ‘the Orcoba class-of-systems’ and ‘the Orcoba-Approach’ interchangeably.

II.4.1 Introduction and research question refinement

This chapter initiates answering the research question that has been presented earlier in this work:

“

Research question 1.1 (Hucolligence with Orcobas). *What are extensions and variants of the Orcoba-Approach that significantly contribute to fostering hucolligence?*

(Quoted from page 48.)

”

Orcoba-extension

The presentation of *some* of these extensions (*‘Orcoba-extensions’*) is at the core of this part. However, the development process of extensions can never be considered to be completed. Therefore it is useful to introduce the following term.

orcoba-fosterer

Definition 3 (*orcoba-fosterer*). *An orcoba-fosterer is someone who fosters the emergence of new extensions and variants of the Orcoba-Approach.* ■

composition-record-user

A complementary notion is the *composition-record-user*, which is a person who uses a specific *composition-record* to acquire or improve the pursued-capability associated with it.

Taking the role of *orcoba-fosterer*, the remainder of this part addresses some of the challenges associated with the aforementioned research question. Below they are formulated as sub-research-questions:

Research question 1.1.1. *[Motivating composition-record-developers] How do you motivate, and continue to motivate composition-record-developers to create new composition-records or improve existing ones?*

[Conclusion(s): page 63, page 116 and page 241] ■

Research question 1.1.2. *[Determining a quality metric for composition-records] How do you effectively measure the quality of composition-records? This is a crucial instrument to determine whether there is indeed progress.*

[Conclusion(s): page 63 and page 117] ■

Research question 1.1.3. *[Dealing with the exploration-exploitation trade-off] How do you balance the trade-off between exploration and exploitation in the development of composition-records?*

[Conclusion(s): page 118] ■

Research question 1.1.3.1. *[Dealing with finite developers effort] The effort of the developers-collective is finite (per time unit), and therefore, has to be spent well. How do you effectively distribute the development effort over the composition-records?*

[Conclusion(s): page 119] ■

Research question 1.1.3.2. *[Dealing with finite assessment resources] The amount of (human and other) resources to assess the quality of composition-records is finite per time unit. How do you distribute these resources effectively over the composition-records?*

[Conclusion(s): page 119] ■

Research question 1.1.3.3. *[Preventing conservatism] How do you prevent too much conservatism? Often, too much development effort is spent on maintaining the status quo, instead of exploring completely new, and possibly more fruitful directions.*

[Conclusion(s): page 120] ■

An example of such conservatism is present in programming language development. (One can consider a programming language design itself as a composition-record.) Among programming language developers it is a well-known problem that popular programming languages evolve very slowly because of conservative forces, such as users being used to the language as it is. This is one explanation for why so many software developers work with antiquated and sub-optimal programming techniques. This was for example the reason that Martin Odersky initiated Scala as a response to the slowly evolving Java programming language (Venners and Sommers, 2009).

Research question 1.1.3.4. *[Increasing equality in treatment of composition-records] How do you give a fair chance to ideas within composition-records that may need time to mature before they lead to a better performance?*

[Conclusion(s): page 121] ■

Research question 1.1.4. *[Dealing with large composition-records] Pursued capabilities that require a long composition-record are problematic in two senses: (1) it becomes very difficult to establish the effect of a relatively small change to the composition-record on the total score of that composition-record, because the longer the composition-record, the larger the sample size needed to show the influence of this change significantly, and (2) players may be more tempted to skip reading parts of the composition-record, which would lead to a lower score without that being an effect of the composition-record. How to deal with these capabilities effectively?*

[Conclusion(s): page 64] ■

The evolutionary theory inspired extensions, treated in chapter II.5 (p. 65) are part of the topic of this chapter. However, because of the lengthiness of the treatment of these extensions, it has been assigned its own, separate chapter. This also means that an important part of research into the research questions posed above, are covered in that chapter.

II.4.2 Classes of systems and extensions

*class-of-
systems*

This section elaborates on my approach to present and explore systems: classes-of-systems and their extensions. A *class-of-systems* is a set of systems with common characteristics. The common characteristics can be expressed as a set of shared design decisions. The notion ‘system design’ as used here includes both the design of a single specific system, and a class-of-systems. In the latter case, it contains a set of design decisions which can be met by more than one concrete system.

*proper-
extension*

Definition 4 (proper design extension). *System design S_E is a proper-extension of system S if: design S_E complies with all design decisions of S , but on top of that S_E complies with additional design decisions.* ■

Example: Suppose S_1 is “A vehicle that moves over land using wheels and can transport at least one person.”, and S_2 is “A bicycle with two wheels with a diameter of 28 inch”. Then S_2 is a proper extension of S_1 . Note that this is very similar to subclass hierarchies in object-oriented programming languages. It is also interesting to note that an extension is both larger and smaller than the system it extends: it is larger in that it has more properties (so its design is longer), but the class-of-systems which comply with the design is smaller (because it contains more conditions to be met)!

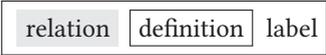
*layered-
system-
approach*

The purposes of explicitly working with system extensions include the following. (1) System extensions enable a *layered-system-approach*: presenting a concrete, detailed, system in a layered, and therefore more understandable, way, by separating the deeper notions from the more concrete detailed implementations: the more general, essential design decisions on top, and the more concrete work out in lower layers. (2) They enable specialisation: for different applications/circumstances, different extensions may be the better (or even only) choice. (3) They enable several types of extensions to be made that are not compatible with each other, but inherit shared properties from a common class (‘ancestor’). (4) They enable optimisation of reusability: separating out design ingredients that are more general to a more general design level with more general objectives, so that potential users of the method can recognise it as applicable to their specific problem. (5) They accommodate dealing in a well-structured way with trade-off relations that may exist between different design options given the objectives (and sub-objectives) of the system: different extensions with a different choice between the trade-offs can be created and compared. (6) They form a good R&D tool: often, it is not clear in advance what extensions will perform best in what situation.

It may be clear that this approach describes explicitly what is involved in any good engineering process, but often only tacitly so.

II.4.3 System-extension overview diagrams

. To render system extensions and properties more comprehensible, this book presents some of them in diagrams. Examples of these diagrams are in fig. II.4.1 (p. 58) and fig. II.4.2 (p. 59). These diagrams are explained as follows:

- Each box represents a system-class or a property that can be attributed to systems of such a class.
- The box connected to the right side of a box, is a *child-box*. The opposite is the *parent-box*. *child-box*
parent-box
-  : each box B can contain up to three pieces of information:
 - **relation** : this indicates the relation between B and its parent-box.¹
 - **definition** : this defines the system-class or the property B represents in terms of other classes-of-systems and/or properties.
 - **label**: this is the label of B . This label briefly describes the system-class or property that B indicates, independent of its relation with its parent and children. In other words, if B is isolation from the rest of the graph, it still should be a good label for indicating the class-of-systems B stands for. B is anonymous when it has no label. This is only allowed if the meaning of B is unambiguously determined by its definition.
- *generic-properties*: this work considers a property (such as ‘eye-colour’) and the value of a property (‘eye-colour = brown’) both as properties. The second is considered to be a sub-property of the first. For example, the property ‘eye colour’ is interpreted as ‘having an eye colour’, while ‘eye colour = brown’ is as having brown eyes. This view is also ‘extensionally’ defensible. ‘Extension’ as used in the latter sentence has to be interpreted in the Carnapian way (Carnap, 1956) and should not be confused with the way ‘extension’ is interpreted in the context of system-extensions. If one defines the Carnap-extension of a property to be all objects that have that particular property, then the Carnap-extension of the property ‘eye colour’ consists of all entities that have an eye colour, while ‘eye colour = brown’ consists of the subset that contains all entities that have a brown eye colour. *generic-
properties*

¹Note that this relation could instead have been represented by an additional box that is on the line between the child-box and the parent-box, and moreover changing the line to an arrow to indicate the direction of an asymmetrical relation. Examples can be found in many RDF-visualisations. However, the chosen representation is more clear, concise and uncluttered.

- Relations:
 - The relation \boxed{P} on a box B with parent B_p means that (1) B represents a property (in the sense of ‘characteristic’, ‘trait’), (2) that B_p represents a system-class, and (3) that the instances of B_p have that particular property.
 - The relation $\boxed{\supset}$ on box B with a parent B_P can have 2 meanings: (1) if B_P represents a system-class, then \supset expresses that B is a strict subclass of B_P . This also implies that the set of instances of B form a strict subset of the set of instances of B_P . (2) if B_P represents a property, then \supset expresses that B represents a sub-property of B_P . *Example: \supset holds when B_P represents ‘eye colour’, and B represents ‘eye-colour is brown’, or when B represents ‘eye-colour is brown or blue’.*
- The definition $\boxed{\text{sysClass1} \cap \text{sysClass2}}$ on box B means that B represents the systems that are in the intersection of sysClass1 and sysClass2 . (Note that none of the names used in the definition are the label of B !)
- \uparrow : this symbol is used on a box B to refer to B ’s parent-box B_P . *Example: Suppose that B contains $\uparrow \cap \text{sysClass1}$. Then B represents the system-class that is formed by the intersection between B_P and the system-class with label sysClass1 .*
- $\boxed{P = V}$ represents a property. To wit: the property of ‘having value V for property P ’. This property is considered to be a subproperty of P , as explained with generic-properties. Note that this implies that the set of all things with property $P = V$ are a subset of all things with property P , to wit those that have value V . An example is formed by choosing $P = \text{‘eye-colour’}$ and $V = \text{‘blue’}$.
- $\boxed{\forall S : \text{prop}}$ represents the system-class of all systems with property prop .

II.4.4 Extensions and generalisations

The Orcoba system-class, as described in chapter II.3 (p. 43), has a number of extensions and generalisations that are relevant to this book. An overview of these is presented in fig. II.4.1 (p. 58). The most general superclass included in this book are Cobas, as introduced in definition 1 (p. 43). Note that throughout this work, Orcobas and not the more general Cobas are regarded as the point of departure. The reason is that Orcobas are the most general class that meet the conditions regarding collective

intelligence systems as pointed out in section I.1.4.1.2 (p. 9). For example, the title of this chapter is not “Extensions of the Coba system-class”.

This book makes a distinction between two types of Orcoba extensions: *general-Orcoba-extensions* and *capability-specific-Orcoba-extensions*.

General-Orcoba-extensions are applicable across a large class of capabilities. This does not imply that each of these extensions can be applied to foster any capability. Some general extensions may only be (effectively) applicable to a large, but more restricted, class of capabilities. The notion ‘general-Orcoba-extension’, therefore, is not a boolean notion (either completely true or completely false), but one with gradations. Capability-specific-Orcoba-extensions, on the other hand, are only applicable to a single or a narrow class of related capabilities.

*general-Orcoba-extension
capability-specific-Orcoba-extension*

A brief explanation of general-Orcoba-extensions developed in, and relevant to this book follows now.

- Challenge-Based-Coba: a Coba in which the quality of a composition-record is estimated by the artificial creation of challenges that have to be dealt with by the composition-record-users. The quality of the solution to these challenges is then used to steer the further evolution of the composition-record(s).
- Challenge-Based-Orcoba: a Coba that is member of the intersection of Orcobas and Challenge-Based-Cobas. In other words, it are Challenge-Based-Cobas that have an open composition-record. These are at the core of this work: they are an essential ingredient in the evolutionary inspired extensions (see chapter II.5 (p. 65)).
- Or-evohut: The class of evolutionary-inspired Orcoba systems are systems that are inspired by evolutionary theory and related technologies. The purpose is to accelerate the development of composition-records to a great degree. They and their extensions are explained in chapter II.5 (p. 65).
- Game-Based-Orcoba: The class of Game-Based-Orcobas consists of Orcobas that motivate humans to participate in it (or its supporting processes) by integrating game elements.

In the context of Game-Based-Orcobas, a composition-record-user is sometimes called a composition-record-player. In other words, this is someone who uses a particular composition-record to improve or acquire the pursued-capability of a given Game-Based-Orcoba.

A brief explanation of the capability specific extensions of the Orcoba system-class developed in, and relevant to this book follows below.

- The SWiFT-class-of-games: this class consists of game-based Orcobas that are specialised for the purpose of fostering formal-fluency. SWiFT stands for 'Semantic Web in Fast Translation'. It and its extensions and variants are further explained in chapter III.6 (p. 215).
- Systems-for-real-time-collective-formal-thinking (systems-for-realcolforthi): these systems foster real-time-collective-formal-thinking. In real-time-collective-formal-thinking a community collaboratively solves a problem, while all members express their thoughts about the problem at the moment they conceive of them in a fine-grained way in a persistent shared digital collective memory (a KR-base) using a formal-language, according to a predefined set of rules. Systems-for-real-time-collective-formal-thinking and their extensions are further explored in chapter III.5 (p. 197).

II.4.5 Challenge-Based-Coba

To measure the performance of a composition-record it is important to define good challenges (tests) for the composition-record-users concerning the capability that the composition-record targets. Challenge-Based-Cobas are defined as Cobas extended with this feature. This section describes what criteria good challenges should meet, and for that purpose also introduces some definitions and notions.

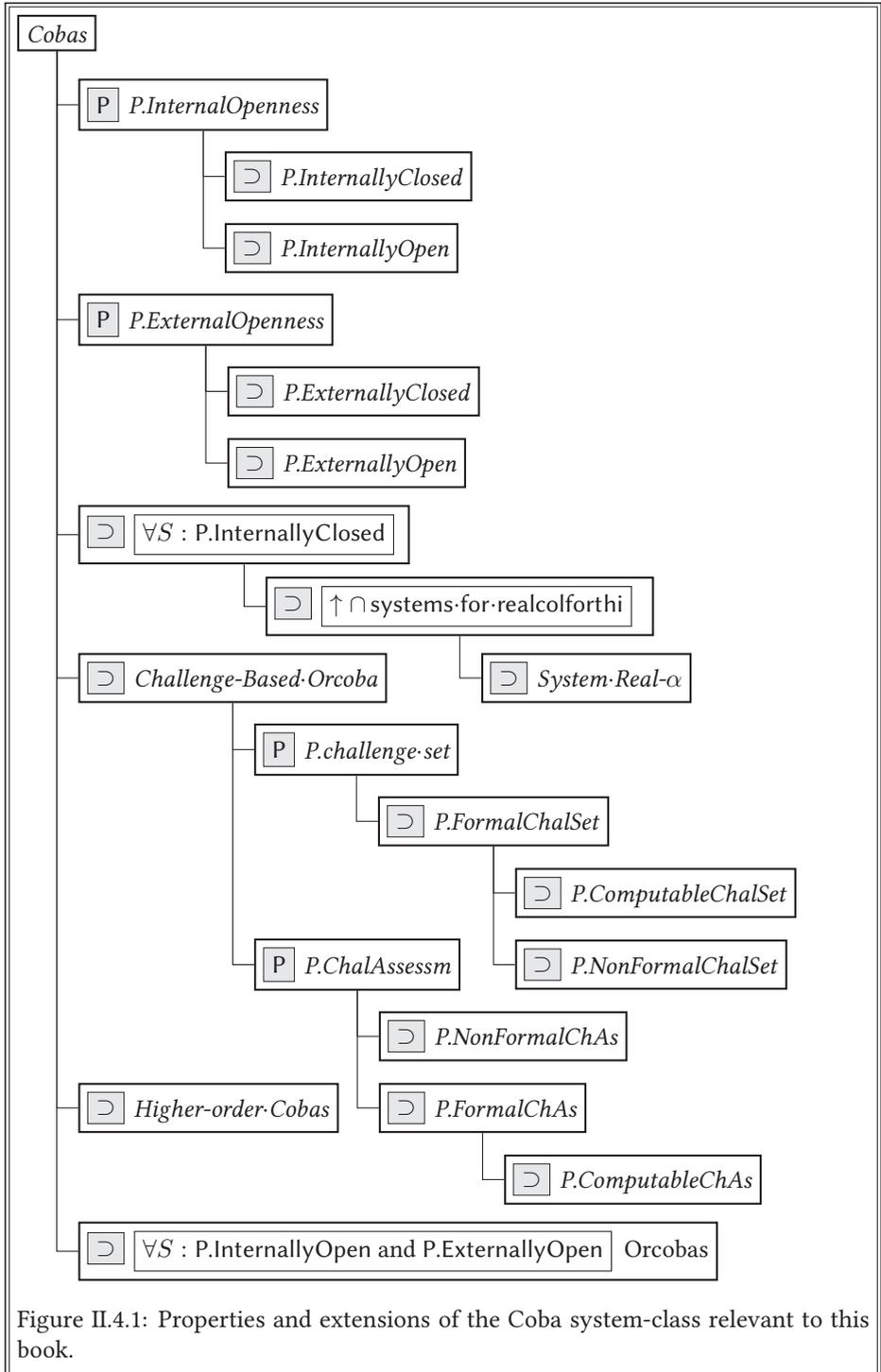
A problem with most descriptions of capabilities is that their meaning is vague. For example, what does it exactly mean to be ‘fluent in English’? Moreover, direct measurement of the degree to which a group or an individual person possesses a certain capability is often impossible, let alone by using an automated procedure. The advantage of vagueness, however, is that is often necessary to conceptualise a given capability most completely – in the sense that the description does not leave out aspects that should be associated with the capability. A description with a greater precision, often sacrifices completeness. For example, equating fluency with speed and accuracy does not do justice to other, sometimes less tangible aspects of fluency (some of which will be described in section III.2.1 (p. 133)). Often, precision and completeness are in a trade-off relation.²

The problem can be overcome considerably by means of a specialisation of Cobas, one that enables accurate and effective assessment of pursued-capabilities: the Challenge-Based-Coba. ‘Effective’ is here used in the sense that the assessment can be done in a reasonable time with the available resources. By developing a sequence of Challenge-Based-Cobas, one can approach effective assessment of the complete, yet vaguely defined capability more and more, without ever reaching it completely. The rest of this section elucidates this and elaborates on the Challenge-Based-Coba.

II.4.5.1 Challenges and challenge-sets

A *challenge* in the context of this book, is a test which has to be faced by the composition-record-user. The quality of the solution of the user is intended to measure the degree to which the user masters the pursued-capability, and the latter degree is used to determine the quality of the composition-record. For example: “What are all solutions to the equation $x + 7^x = 15$?”. A *challenge-set* is a set of challenges *challenge-set* in the mathematical sense, except that the criterion of membership to the set can be non-formal, and even non-formalisable. Challenge-sets are important for several reasons. While assessing the performance of a user, multiple challenges will mostly be necessary to get a reliable assessment. Moreover, an ongoing addition of new challenges is needed to minimise copying of solutions between users (compare it

²Section III.3.2.2.1 (p. 156) elaborates on this in much more detail.



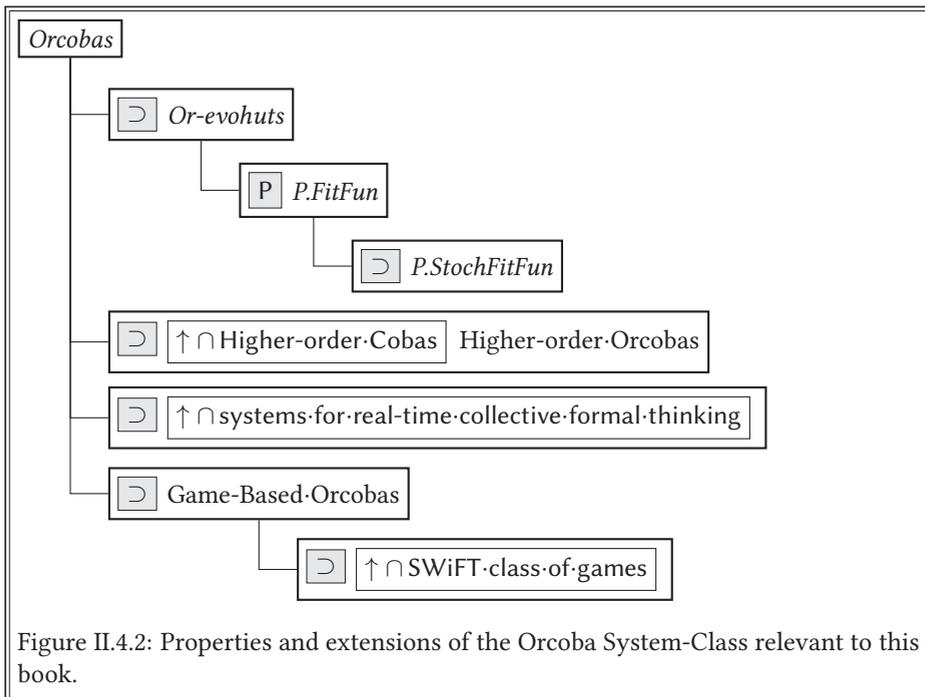


Figure II.4.2: Properties and extensions of the Orcoba System-Class relevant to this book.

with giving the same exam over and over: sooner or later the students find out, and are going to pass the answers to others).

A challenge-set is *ideal for a capability* C , if it exactly coincides with the set of all challenges that form an excellent test for the pursued-capability. An ideal challenge-set is intended to capture the notion of the capability in a measurable way, i.e. scoring well on most instances of the challenge-set implies the capability is mastered. The problem with challenge(-sets) is that it may be a labourious task to create the challenges and to assess the solutions of users. For this purpose, we define formal and computable challenge(-sets), which allow computers to assist in the process.

*formal-
challenge*

A *formal-challenge* is a challenge with a mathematically formal way to assess the quality of the solution to it. A *formal challenge-set* is a mathematically definable set of formal-challenges. Hence, this set is formal in two ways: both the set and the elements have a formal nature. *Example: The set of all equations with one unknown, in which only the operators $+$ and \times occur, such as $x+7 \times x = 15$, each joined with the question: "What are all solutions to this equation?"* A stronger and more restrictive notion to formal challenge-sets are computable challenge-sets. A computable challenge-set is a set which can be enumerated by a computer program, or, to be more precisely: by a Turing machine.

Non-formal-challenges and non-formal challenge-sets are to be contrasted with formal-challenges and challenge-sets. A non-formal challenge-set is a set of challenges that is not formally defined. That is, the membership to the set is and cannot be specified mathematically. *Example: The set of equations that can be easily solved by the average person with an MSc. degree.* Note, that the latter example shows that the elements of a non-formal challenge-set can be formal-challenges. An example of a non-formal challenge-set with non-formal-challenges is "the set of drawings of human faces". Interestingly, these can be assessed by humans fairly well. Non-formal-challenge and non-formal challenge-sets also serve an important purpose, that has already been hinted at by using the strongly related notion of 'vagueness' in the introduction of this section. This is because many capabilities will often themselves be quite non-formal(isable) notions, and determining the pursued challenge-set is an ongoing quest of refinement. The more relaxed notion of non-formal-challenges, will allow better approaching the ideal challenge-set (if it exists).

*artificial-
creatability*

Artificial-creatability of a challenge-set is the degree to which the challenges of a challenge-set can be created 'artificially'. (This includes the question of whether it is ethical to do so.) The latter does not necessarily mean 'by a computer'. It is about the requirement that the challenges of the challenge-set can be created on demand, at any time one wishes. For example, a challenge-set consisting of 'resolving human conflicts' has a low artificial creatability: conflicts cannot easily be created artificially, and moreover it would even be unethical to do so!

II.4.5.2 Challenge-assessment

A *challenge-assessment* is the assignment of a quality score to the solution of a challenge, i.e. a function from the solution of a challenge to quality score. Just as with challenges this challenge-assessment can be non-formal, formal, or even computable. Moreover, the quality score itself can be quantitative or qualitative. The score can, moreover, have the property of *averageability*: there is a (computable) function from a set of quality scores to a single score, which represents the average performance on challenges.

II.4.5.3 Designing challenge-generation and assessment systems

Challenge-creation and challenge-assessment can be carried out by humans or by computers, or by using a hybrid approach. What is best to do depends on what the options are and how efficient they are to implement given the resources available to the orcoba-fosterer. Examples are:

- If the challenge-generation or challenge-assessment can be carried out effectively by a computer program, which in its turn can be programmed with reasonable effort within reasonable time, this will in most cases be more efficient than that it would be done by humans, and is therefore preferred.
- If this is not possible, humans can carry out the challenge-generation and/or challenge-assessment.
- If the pursued challenge-set is non-computable or even non-formalisable, it may be possible to approach it with an ever-growing sequence of computable challenge-sets. Each challenge-sets is designed by humans as an improvement of the previous set. A concrete example is mathematical proving, so, given a set of axioms and a statement, prove that the statement, or its negation follow from the set.

II.4.6 Higher-order-Orcobas

Higher-order-Orcobas form the answer of this work to research question 1.1.4 (p. 51) (dealing with large composition-records). Instead of tackling the pursued-capability with one 'flat' Orcoba, a higher-order-Orcoba is defined in terms of a set of other Orcobas, each of which focus on a subcapability that is easier to grasp in isolation. More specifically, higher-order-Orcobas work as follows. (1) Break up the complex capability into smaller subcapabilities. Choose the subcapabilities such that a good

sub-Orcoba

composition-record can be written which occupies at most a given maximum space (for example: 300 characters). This maximum space is a hard limit: composition-record game players are not allowed to create larger composition-records. Each subcapability is targeted with its own *sub-Orcoba*. (2) Define dependencies between these capabilities. A player only gets access to a sub-Orcoba that focuses on a subcapability, after he masters the capabilities on which it depends to a sufficient degree. (*End of list.*)

The last point is important, because it allows for modularity in the performance assessment of composition-records. Ideally, one wants to be able assess the performance of the composition-records of a given sub-Orcoba S only for the subcapability c that S is defined to target. Suppose that a composition-record r of S , which should only target capability c , also illicitly contains parts that target subcapabilities that already have been targeted by other sub-Orcobas. If the player masters the subcapabilities targeted by the other sub-Orcobas to a sufficient degree, then the illicit parts of r do not influence the score of the player of S . In other words, the modularity of the assessment is not in danger. *Example: Consider a design for a higher-order-Orcoba O for learning how to ride a bicycle. It consists of two sub-Orcobas. Sub-Orcoba S_1 focuses on physical mastery of cycling, so how to pedal, keep balance and steer. Sub-Orcoba S_2 focuses on how to obey the traffic laws while riding the bicycle. In O , the second subcapability is defined as being dependent on the first subcapability. Now, suppose that a person does not yet physically master cycling, but is already admitted to S_2 . Then a composition-record for S_2 that also helps him to improve his physical mastery of cycling, will positively influence his score. It may even beat a composition-record that is actually better in helping him to improve the targeted capability of obeying the traffic laws. This situation is in conflict with the purpose of S_2 . This situation does not occur, if the person physically masters cycling before being admitted to S_2 .*

*dependent-
capability-
graph*

The term *dependent-capability-graph* is used for the mentioned structure of subcapabilities and dependencies between them.

Higher-order-Orcobas form indeed a possible solution to research question 1.1.4 (p. 51). First, the smaller a composition-record, the greater the effect of changes to parts of it. Second, it is easier for a player to read it completely.

II.4.7 Conclusion

This chapter dealt with a part of the research questions posed in its introduction (section II.4.1 (p. 49)). The following repeats these research questions, including a summary of the results obtained so far. These and the remaining research questions are further explored in a later chapter: chapter II.5 (p. 65).

“

Research question 1.1.1 (motivating composition-record-developers). *How do you motivate, and continue to motivate composition-record-developers to create new composition-records or improve existing ones?*

(Quoted from page 50.)

”

Conclusion 1 of research question 1.1.1. *This section has presented Game-Based-Orcobas to spur people to participate in an Orcoba (see section II.4.4 (p. 54)).* ■

“

Research question 1.1.2 (determining a quality metric for composition-records). *How do you effectively measure the quality of composition-records? This is a crucial instrument to determine whether there is indeed progress.*

(Quoted from page 50.)

”

Conclusion 1 of research question 1.1.2. *The Coba-extension Challenge-Based-Coba was introduced for this purpose. It requires the definition of good challenges (tests) for people using it. The (average) score of a sufficient number of people on a sufficient number of challenges per person, is used as a score for the composition-record. For more details, see section II.4.5.1 (p. 57).* ■

“

Research question 1.1.4 (dealing with large composition-records). *Pursued-capabilities that require a long composition-record are problematic in two senses: (1) it becomes very difficult to establish the effect of a relatively small change to the composition-record on the total score of that composition-record, because the longer the composition-record, the larger the sample size needed to show the influence of this change significantly, and (2) players may be more tempted to skip reading parts of the composition-record, which would lead to a lower score without that being an effect of the composition-record. How to deal with these capabilities effectively?*

(Quoted from page 51.)

”

Conclusion 1 of research question 1.1.4. *For this purpose higher-order-Orcobas have been introduced (see section II.4.6 (p. 61)). Instead of tackling the capability with one 'flat' Orcoba, a higher-order-Orcoba is defined in terms of a set of other Orcobas, each of which focus on a subcapability that is easier to grasp in isolation, and that requires a much shorter and easier digestible composition-record. ■*

Chapter II.5

Extensions inspired by evolution theory

(Before starting this chapter, I would like to gently remind the reader of the call for feedback on page i.)

II.5.1 Overview and structure

One of the essential ingredients of the Orcoba-Approach consists of *evolutionary systems* as a means to effectively develop good composition-records for specific capabilities. These systems are inspired by aspects from evolutionary biology (Darwin, 1909; Gould, 2002), and are related to the modern application of this theory in A.I.: evolutionary computation. This section presents the theory behind these evolutionary systems and specific designs and design considerations, each inducing specific extensions of the Orcoba-Approach.

Before starting the explanation of the designs, required foreknowledge is presented. General *existing* evolutionary computation theory is presented (section II.5.3 (p. 67) and section II.5.4 (p. 71)), and after that a selection of specific existing techniques that are applied in this work (such as Sigma scaling) are explained (section II.5.5 (p. 73)).

Subsequently, a generalisation is made from evolutionary computation systems to evolutionary human-technological systems (EvoHuts) (section II.5.6 (p. 76)).

The section concludes with the central contribution from the perspective of the Orcoba-Approach: a specialisation of EvoHuts for Orcobas: ‘Or-evohuts’, and presents a specific design (section II.5.7 (p. 110)).

II.5.2 Introduction and research questions

As has been stated in section II.4.1 (p. 49) the Orcoba-Approach (see chapter II.3 (p. 43)) deals with one of the main research questions of this book (research question 1) by satisfying this work's main principles for fostering hucolligence predominantly by 'definition'. Making a system that actually *implements* these principles, gives rise to several sub-research questions, that are non-trivial to solve. These questions have been presented in section II.4.1 (p. 49). Some results have already been presented in chapter II.4 (p. 49). This chapter continues with the most elaborate and central result: the evolutionary inspired extensions.

One of the essential elements of the evolutionary extensions is, among other things, inspired by aspects from evolutionary biology (Darwin, 1909; Gould, 2002), and is related to the modern application of this theory in Artificial Intelligence: evolutionary computation (EC). In an analogy to biological evolution, composition-records can be seen as living individuals, which, depending on their success, have a greater probability to attract resources (mainly human composition-record developers) for producing offspring, which are typically variations on themselves. Some of these variations may even be more successful than their parents, and this sparks a process of gradual improvement. The variations in the composition-records are caused by humans editing the composition-records, and success is defined by the average performance of humans using the composition-record.

Applying ideas from evolutionary theory effectively in a concrete Orcoba system is a rather complicated matter. The purpose of this chapter is to introduce a general class of such systems ('Evhuts'), and specific well-defined evolutionary Orcoba-extensions

As said, the systems presented are *partly* inspired by the mentioned theories and technologies. There are some notable differences. Most of the systems differ from biological evolution in a number of ways, including:

- In all systems presented, variation is not produced by random biological processes normally associated with Darwinistic evolution, but by conscious design by human beings.
- In all systems, there is no 'auto-influence' of individuals on what it means to be fittest (among others through a process called co-evolution). In contrast, in biological evolution, biological individuals influence and change their environment including other species, in their turn changing what it means to be the fittest. *Example: a plant produces oxygen, which in its turn changes the environment, which makes it possible for other species, such as lice, to come into existence, which in their turn interact with these plants, causing the property of 'being sufficiently resistant to lice' a new factor in being fit.* This is an example

of two co-evolving species. In the systems in this section (and in general in general evolutionary computation systems), however, the fitness is defined external to the individuals.

Moreover, all or some of these systems differ from archetypical evolutionary computation in the ways described in the following. (Note that there may exist non-typical forms of evolutionary computation that share some of these ways.)

- In all systems, variation is not produced by computer programs, but by deliberate design by human beings. However, it may be complemented with random mutation operators similar to what is common in evolutionary computation.
- In most systems, the fitness evaluation is (at least partly) carried out by human beings, and can at most be partly complemented with evaluation by computers.
- The systems transcend the computer-technological aspect to integrate human activity in it. For this, I coin the term *evolutionary human-technological system*, and the more restrictive notion *evolutionary human-computational system*.
- In some systems entities live indefinitely long, and all offspring are proper variations on (so not clones of) their parents. Note that this effect is not mimicked by creating a clone of an individual: normally a clone will get a ‘clean slate’ when its fitness is evaluated, while in some of the systems here, it is really about the same individual: the ‘karma’ (fitness evaluation information) will be integrated in the fitness evaluation at a later stage.
- The population size is not fixed in all systems, in contrast to most evolutionary computation systems.

II.5.3 Introduction to evolutionary computation

This section presents an introduction to existing general evolutionary computation theory. Evolutionary computation is defined as the field which uses ideas inspired by biological evolution (Darwin, 1909) for the design of computer programs, which are capable of automatically and effectively generating good solutions for a given problem. Related terminology, relevant to this book, consists of: evolutionary algorithms (EA), genetic algorithms (GA), genetic programming (GP) and human-based evolutionary computation.

The terminology used within the field of evolutionary computation, including the name of the field itself, is not univocal among the associated communities of researchers. For example, Simon states that a sharp definition is non-existent, and

states that “a genetic algorithm is one, if it is generally considered a genetic algorithm” (Simon, 2013, p. 3), which either indicates that the boundaries are vague and cannot be defined properly, or the terminology used by different communities has not sufficiently converged. Schoenauer and Michalewicz (1997) point out that: “very often the terms evolutionary computation methods and GA-based methods are used interchangeably. Is it because of their fashionable name and concepts, or due to better marketing policy of the corresponding community?”. Eiben and Smith (2003) state that the differences between the fields are mainly historical and representational and not essential. These fields have mainly chosen different representations for population-individuals and other selection operators. However, essentially they are equally expressive. Depending on the problem, one representation may feel more ‘natural’, while with some effort, it could also have been represented in another subfield. This situation may be similar to that in another field: formal models for computation, of which the most famous are: lambda-calculus, Turing machines and recursive functions. They are equivalent, but, indeed depending on what one wants, one representation can be easier to use than the other. Eiben furthermore states: “Today there are advanced versions of Genetic Algorithms, Genetic Programming, Evolution Strategies, Evolutionary Programming, [...]. The lines between these flavors of evolutionary computation are blurred to the point where it is difficult to pinpoint where one system ends and another begins.” and “As you can see, everything is pretty much blurred together, a few modifications to your algorithm, modifications that simply make it easier to encode the genotype for some particular project you are working on, and there will be those who will start calling your evolutionary algorithm approach by a new name.”

In this book, the definition used by Eiben (and many other specialists in the field) will be used: evolutionary computation is the umbrella term for all evolutionary inspired A.I. systems, which, moreover have the general design structure described in section II.5.3.1, or a very similar one, in common. There is no need for referring to a specific approach within this field, because what they have in common is what is relevant to this book.

II.5.3.1 Definitions and properties

There are several notions, definitions and properties from the area of evolutionary computation that are relevant to the systems in this book. They are briefly explained as follows.

The main ingredients of an evolutionary computation system are

- Parent selection
- Variation operators (mostly mutation and recombination operators)

- Fitness evaluation
- Offspring selection

Choosing concrete algorithms for each of the above, results in a specific evolutionary algorithm.

The overall evolutionary computation-algorithm ‘blue print’ is as follows (Eiben and Smith, 2003):

```
BEGIN
INITIALISE population with random candidate solutions;
EVALUATE each candidate;
REPEAT UNTIL ( TERMINATION CONDITION is satisfied ) DO
1 SELECT parents;
2 RECOMBINE pairs of parents;
3 MUTATE the resulting offspring;
4 EVALUATE new candidates;
5 SELECT individuals for the next generation;
OD
END
```

Each ‘cycle’ through the algorithm represents a *generation cycle*, or equivalently an *iteration* (of the evolutionary computation-algorithm). *generation cycle*
iteration

Further important notions and terminology are the following.

- *Solution space*: the collection of possible solutions to the problem. *solution space*
- *Representation space*: in most evolutionary computation systems, the solutions are not directly manipulated (by variation operators), rather their *representations* are. *representation space*
- *Population*: a subset of the representation space, representing currently ‘living’ individuals. *population*
- *Phenotype*: members of the solution space are often called phenotypes. This is in an analogy to biological systems, in which the phenotypes are a direct expression of the properties of an individual, which determine their success at reproducing. *phenotype*
- *Genotype*: members of the representation space are often called genotypes. In an analogy to biological systems, the genotypes map to phenotypes, and are the target of variation operators. *genotype*

- fitness evaluation*
- *Fitness evaluation*¹: one *fitness evaluation* (or *fe*): determining the fitness of one individual from the population, equivalently: applying the fitness function to an individual.
- fitness score*
- *Fitness score* or *score*: The outcome of a fitness evaluation.
- selection operator*
- A *selection operator* is the part of the evolutionary computation-algorithm that selects the individuals for reproduction.
 - Major categories of selection operators are, amongst others, fitness proportionate selection, rank based selection and tournament based selection.
- fitness proportionate selection*
- In *fitness proportionate selection* the probability that an individual is selected to become a parent is proportionate to its fitness.
- rank based selection*
- In *rank based selection* the individuals of a population are ranked according to their fitness.
- tournament based selection*
- In *tournament based selection* subsets of the population are chosen to play tournaments. The outcome of the tournament determines the probability that a competing individual is chosen to become a parent.

The terminology regarding phenotypes and genotypes as used in evolutionary computation can be a bit confusing, and does not completely correspond with the way these terms are used in biology. In biology, an *individual of the population* has a phenotype and a genotype. In evolutionary computation, the term individual of the population is often used as a synonym for a member from the representation space, and thus the genotype.

II.5.3.2 Design considerations

- takeover time*
- Designing a good evolutionary computation system is not trivial. Therefore, a reflection on design considerations is useful. The following first introduces the most important notions to be able to express these considerations. The *takeover time* is the speed at which the best solution in the initial population would occupy the complete population by repeated application of the selection operator alone, thus, without applying variation operators (Bäck, 1994; Goldberg and Deb, 1991). The takeover time is closely related to the *selective pressure*, which conceptually represents the degree of importance to have a high fitness in order to produce offspring. A way to formalise selective pressure is expressing it as the inverse of the takeover time. The
- selective pressure*

¹This may not be general terminology from the area of evolutionary computation.

disruptiveness of the variation operators is the degree to which variation operators introduce changes in the offspring.

Some important design considerations are as follows. The shorter the takeover time (equivalently: the higher the selective pressure), the quicker the variation in the population will decrease, with an increased danger that the system will end up in a local optimum further below the global optimum than with higher takeover times. This can be compensated in at least two ways: (1) By applying a higher disruptiveness of the variation operators. (Note that changes in the variation operator do not change the takeover time, because the latter is defined in terms of the selection operator *alone*, so assuming that the variation operators would not be applied.) (2) By a larger population size. (*End of list.*)

II.5.4 Definitions from general evolutionary computation

This section supplements the general introduction into evolutionary computation (section II.5.3 (p. 67)) with formal definitions and notations.

The *objective fitness function* F_o represents the quality measure for different solutions: *objective fitness function*

$$F_o : \mathcal{S} \rightarrow \mathcal{R}_o, \quad (\text{II.5.1})$$

where \mathcal{S} is the *solution space* (also search space), and \mathcal{R}_o represents the objective *solution space*
score of a solution. The solution space contains all possible genotypes.

In fitness proportionate selection the fitness values assigned to individuals by the objective fitness function are in most cases not suitable to be used without some additional transformations. A typical situation where this may be the case is when a population starts to converge towards an optimum. The difference between the objective fitness of the individuals within the subsequent population will become smaller and smaller. A selection operator that is directly based on the global fitness function may therefore have a takeover time that is larger than desirable (in other words: its selective pressure is lower than desirable).

This is solved by transforming the objective fitness function. In other words, the fitness function used for the selection algorithm will be defined in terms of the objective fitness function, to which some transformations are applied. A typical class of such transformation functions are scaling functions. In the case just given, these will ‘exaggerate’ the differences by scaling the objective fitness function on a domain limited to the current population. Exact definitions follow later.

This work uses the term *selective fitness function* for the latter function, distinguishing it from the objective fitness function. The addition ‘selective’ indicates *selective fitness function*

that it is the fitness function used to carry out the actual *selection* process in the evolutionary computation system.² In this book a distinction is made between a global selective fitness function and local selective fitness functions.

Definition 5 (global selective fitness function). *Given an objective fitness function F_o , then a function Φ_g with the properties defined below, is a global selective fitness function:*

$$\Phi_g : \mathcal{S} \rightarrow \mathcal{R}_f, \quad (\text{II.5.2})$$

fitness-
evaluation-
space

where \mathcal{R}_f is a set, mostly \mathbb{R}^+ , which represents the fitness-evaluation-space. Moreover, Φ_g shares its maxima with either (1) the maxima of F_o , if one wants to optimise F_o , or (2) with the minima if one wants to minimise F_o (end of list). ■

However, the notion of a global selective fitness function will not suffice for the selective fitness functions of interest in this book, because the value of the selective fitness function may depend on the composition of the given population.³ In other words, the fitness evaluation of an individual may change if you change the phenotypes of the other individuals in the population. This is particularly the case when applying fitness scaling. The local selective fitness function and its generator are introduced for this purpose:

Definition 6 (local selective fitness function). *Given a finite sub-multiset of \mathcal{S} , $P = \{s_1, s_2, \dots, s_n\}$, then Φ_P is a local selective fitness function if the following holds.*

$$\Phi_P : P' \rightarrow \mathcal{R}_f \quad (\text{II.5.3})$$

Where P' is the (normal) set that contains the same elements as multiset P (i.e. iff an element occurs one or more times in P , it occurs exactly once in P'). Moreover, Φ_P should preserve the ordering imposed by the objective fitness function F_o . In other words, $F_o(s_1) < F_o(s_2)$ iff $\Phi_P(s_1) < \Phi_P(s_2)$. ■

²Note that in the literature, slightly different terminology is used: *objective function* (instead of objective fitness function) and *fitness function* (instead of selective fitness function), respectively. However, I deem the latter less intuitive. Both functions are fitness functions, however, the objective fitness function provides the fitness from the perspective of the user of the endproduct, while the selective fitness function provides the fitness from the perspective of the selection algorithm. In other words, both functions can be regarded as specialisations of a more general notion of fitness function. The word ‘objective function’ does not express this. However, with the terminology used here, the word-combination ‘fitness function’ can now be used when it is not needed to make a distinction between the objective fitness function and the selective fitness function, or when the context makes it clear which one of the two it is.

³Note that there are also selective fitness functions that even have more dependencies, such as dependence on the worst value over the last few generations (Bäck et al., 1997, p. C2.2:1).

Definition 7 (local selective fitness function generator). *A local selective fitness function generator Φ_{loc} is a higher-order function with the following type:*

$$\Phi_{loc} : \mathcal{P}_{fm}(\mathcal{S}) \rightarrow (D \rightarrow \mathcal{R}_f), \tag{II.5.4}$$

where $\mathcal{P}_{fm}(X)$ is the set of all finite sub-multisets of X (so, a finite multi-set, which exclusively contains elements from X), and where the domain D of the resulting function is equal to the parameter offered to Φ_{loc} . The resulting function, $\Phi_P = \Phi_{loc}(P)$ must further be a local selective fitness function. Notation: $\Phi_{loc.P}$ is an alternative way to write $\Phi_{loc}(P)$. ■

The (non-formalised) semantics of these expressions is as follows: P represents a population of individuals at a specific generation cycle of the evolutionary computation-algorithm, and $\Phi_{loc}(P)$ produces a selective fitness function for this population.

Now we have everything at our disposal to define the proportional selection scheme.

Definition 8 (class-of-proportional-selection-operators). *Given are a finite multi-set $P \subset \mathcal{S}$ (representing a population at a certain iteration), and a local selective fitness function generator Φ_{loc} . The class-of-proportional-selection-operators based on the given information, consists of all operators for which the probability P_{prop} that an individual $s \in P$ is chosen to become parent is*

$$P_{prop}(s) = \frac{\Phi_{loc.P}(s)}{\sum_{s \in P} \Phi_{loc.P}(s)}$$

Note that the equation in definition 8 does not uniquely define a *specific* operator, this may be counter to the first intuition of some readers. In general, a predefined probability distribution does not yet uniquely define the probabilistic system that generates such a distribution. In section II.5.5.1 on the next page an example of a fully specified proportional selection operator is given.

II.5.5 Existing techniques within evolutionary computation

The evolutionary human-computational system described in later sections applies some specific existing evolutionary computation techniques, to wit the so-called

proportional selection scheme, Sigma scaling and some associated notions. This section defines these in precise terms.

II.5.5.1 Proportional selection operator: stochastic universal sampling

The following defines a selection operator that complies with definition 8 on the preceding page: Stochastic Universal Sampling (SUS) (Baker, 1987). SUS has a few advantages compared to some other operators. First, the variation from the expected value is not (too) great. For example, an individual with a high probability will always produce offspring. Moreover, it runs efficiently. SUS can be visualised as follows. Imagine a wheel with as many spokes as the number of children to be produced. The spokes are equally spaced. Moreover, imagine a roulette wheel in which each population individual takes an area proportional to its selection-probability. Place the spoked wheel over the roulette wheel, so that the spinning axes of both wheels coincide. Moreover, assume that the roulette wheel is fixed to the table and cannot spin, in contrast to an ordinary roulette wheel. Spin the spoked wheel and wait until it rests again. The number of spokes that hover over the roulette wheel area of an individual, is the number of assigned children to that individual. The formal definition, in terms of an algorithm, is as follows.

Definition 9 (Stochastic Universal Sampling). *Stochastic Universal Sampling is defined by the next algorithm (Baker, 1987).*

```

SUS(Population, N) =
{  TotalFitness := total fitness of population
   NumberOfChildren := number of offspring to keep
   PointerDistance := distance between the pointers
                       (= TotalFitness/NumberOfChildren)
   Start := random number between 0 and PointerDistance
   Pointers := [  Start + i*PointerDistance
                 | i in [0..(NumberOfChildren-1)]
                 ]
   return FitnessProportionateSelection(Population, Pointers)
}

FitnessProportionateSelection(Population, Pointers) =
{  ParentSelection = []
   i := 0

   for each Pointer in Pointers do

```

```

{   while ( fitness sum of Population[0..i] < Pointer ) do
    { i++ }
    add Population[i] to ParentSelection
  }
return ParentSelection
}

```

■

Where $\text{Population}[0..i]$ is the set of individuals with array-index 0 to (and including) i .

The gist of the above algorithm is that the `while`-loop is skipped as long as the `Pointers` selected in the outer for loop lie before the border to the next roulette-wheel area (which belongs to the next population individual). By skipping it, the index i remains constant, and thus the individual is added a number of times to `ParentSelection` that is equal to the number of `Pointers` that ‘hover’ above the roulette wheel area of that individual.

II.5.5.2 Selective fitness function: Sigma scaling

Finally, we need to define a specific selective fitness function in terms of the objective fitness function. Among many other methods, *Sigma scaling* is a way to create such a selective fitness functions (Hancock, 1994). It is defined as follows.

Definition 10 (Sigma scaling). *A selective fitness function is the result of Sigma scaling of an objective fitness function F , if it can be expressed as follows:*

$$\Phi_P(s) = F_o(s) - \left(\overline{F_o(P)} - c\sigma(F_o(P)) \right) \quad (\text{II.5.5})$$

Where $F_o(P)$ is the multi-set that results from applying function F_o to all elements of P , and c is some constant. Moreover, given a multi-set X , \overline{X} is the sample mean of X , and $\sigma(X)$ is the sample standard deviation of X . ■

Sigma scaling is included here because it is part of the system Or-evohut- α described later in this chapter. The reader should, however, not attribute too much weight to this design decision. Future work and experimentation has to point out which form of fitness scaling is most appropriate, or whether it is better to choose for a selection operator that does not require scaling. Therefore an explanation of the precise functioning of Sigma scaling is omitted.

II.5.6 Evolutionary human-technological systems (Evohuts)

Essential to the approach of fostering hucolligence in this book are what I have coined the class of Evohuts (evolutionary human-technological systems), which are human-technological systems inspired by evolutionary theory. Both activity of humans and activity of technologies are first-class citizens in the design of these systems. From the perspective of evolutionary computation, Evohuts can be regarded as an extension of evolutionary computation systems. In other words, the class of Evohuts are a *superset* of evolutionary computation – so all systems for the latter are a subset of these systems. Related notions are human-based genetic algorithms (Kosorukoff, 2001) and interactive evolutionary computation (Pei and Takagi, 2018; Takagi, 2001).

II.5.6.1 Orcoba systems and human-capability-enhancing-Evohuts

The Evohuts central to this part of the book are a collection of Orcoba evolutionary systems, i.e. extensions of the Orcoba-Approach that integrate an Evohut. Another way to phrase it, is that they form the intersection of the class of Orcobas and Evohuts. I coin the term *Or-evohut* for this class. The dash is placed to enforce a certain pronunciation. It stands for Orcoba Evolutionary Human-Technological System.

Note that most design ingredients of these Or-evohut systems are more generally applicable. In other words, they may be applicable to a wider class of Evohuts than to Or-evohuts only. If applicable, the design ingredients presented in this chapter are presented in a way that reflects this. The following introduces a more general system-class for this purpose. In this way, it may be easier to see how to apply them beyond the scope of Orcoba systems.

A *human-capability-enhancing-Evohut* is an Evohut in which (1) the *evo-individuals* (the individuals of the solution space) consist of some sort of method to create conditions for human subjects to acquire and improve certain collective or individual capabilities, such as training methods and protocols, and (2) the performance-level of these capabilities can be suitably represented by a so-called performance-function (*end of list*). These functions are introduced in section II.5.6.2 on the next page . Moreover, the *performance-level*, in the context of this book is, ideally, the degree to which subjects are capable to perform with respect to a certain capability, under the expected conditions. It applies to a particular point of time, in contrast to some performance notions introduced later.

Or-evohut

human-capability-enhancing-Evohut
evo-individual

performance-level

II.5.6.2 Types of objective fitness functions in Evohuts

In Evohuts, there are two options for defining fitness functions that do not exist in evolutionary computation. Next to automated fitness evaluation, as in evolutionary computation, pure human evaluation and hybrid human-algorithmic evaluation are possible as well.

A subclass of Evohuts with *purely human fitness evaluation* consists of those that employ direct conscious human judgment of evo-individuals, for example an Evohut in which the evo-individuals are drawings that have to be judged on aesthetic qualities. *purely human fitness evaluation*

In *human-technological fitness evaluation*, the fitness evaluation is defined in terms of a mix of human and algorithmic activity. A subclass of Evohuts in which this is the case, is formed by Evohuts in which the evo-individuals consist of some sort of method to train and/or guide human subjects carrying out certain tasks. *Example: an Evohut, in which the evo-individuals are formed by training-methods for basketball-teams. A suitable fitness score, in this example, is one that depends statistically (the algorithmic part) on the score of the humans using the method (the human part).* That means that, in this case, the score does not express the direct opinion of human subjects about the quality of the evo-individual, but the more objective effect of the evo-individual on the performance of these human subjects. *human-technological fitness-evaluation*

A number of types of fitness functions and related notions are introduced in this section, specifically those that are important in the capability-enhancing Evohuts of specific interest in this book (Orcobas and the SWiFT game): stochastic objective fitness functions, objective fitness functions based on dynamic performance, and stochastically measured dynamic performance.

II.5.6.2.1 Stochastic objective fitness functions

In some cases, an objective fitness function has a stochastic nature. This is the case if the objective fitness function is intended to estimate a statistical parameter. The value one would like to define as the objective fitness function cannot be accessed directly, but only estimated statistically. Typically, such an estimation is based on the average of a sample of individual measurements.⁴As a consequence, the truly pursued value can only be measured to a certain degree of precision. This also implies that determining the completion of an evaluation becomes a subjective process: it depends on how much precision you want. The greater the sample size, the greater the precision. An undesirable property of stochastic objective functions is that they

⁴Stochastic fitness functions are not to be confused with stochastics applied in the selection algorithm, such as in Stochastic Universal Sampling. In SUS, the objective fitness function does not have to be a stochastic fitness function, it is merely the selection algorithm itself which is of a stochastic nature.

contribute to more costly evaluations. The more precision you want, the more costly the evaluation.

Figure II.5.1 on the facing page contains an example from the SWiFT game, a game which belongs to the class of Evohuts. The individuals of the population are composition-records, and their fitness score is based on the average of the scores of the players who used the composition-record to play the game. In other words, the random variable associated with each composition-record is the score of a player who used that composition-record to know how to play the game, and the fitness evaluation is based on a sample of this random variable.

Note that this example can also be used to illustrate the reason for the enhanced terminology as introduced in section II.5.6 (p. 76): in this case it prevents confusion over some overlapping terminology between evolutionary computation and statistics. Both employ the word ‘population’ and ‘individual’, however, in different senses. In this particular example, the population of the Evohut consists of a set of composition-records, while the population from which the statistical sample is drawn is the set of scores that composition-record-users achieved using a particular composition-record. When needed, the terms are disambiguated by prefixing ‘evo-’ to the word, indicating that the term has to be interpreted from the perspective of evolutionary computation. An example of a term which has already been used is evo-individual, an individual of the solution space of an Evohut. Note that in the sequel the terms composition-record-user and composition-record-player are used interchangeably, because a great part of this study was developed in the scope of Game-Based-Orcobas.

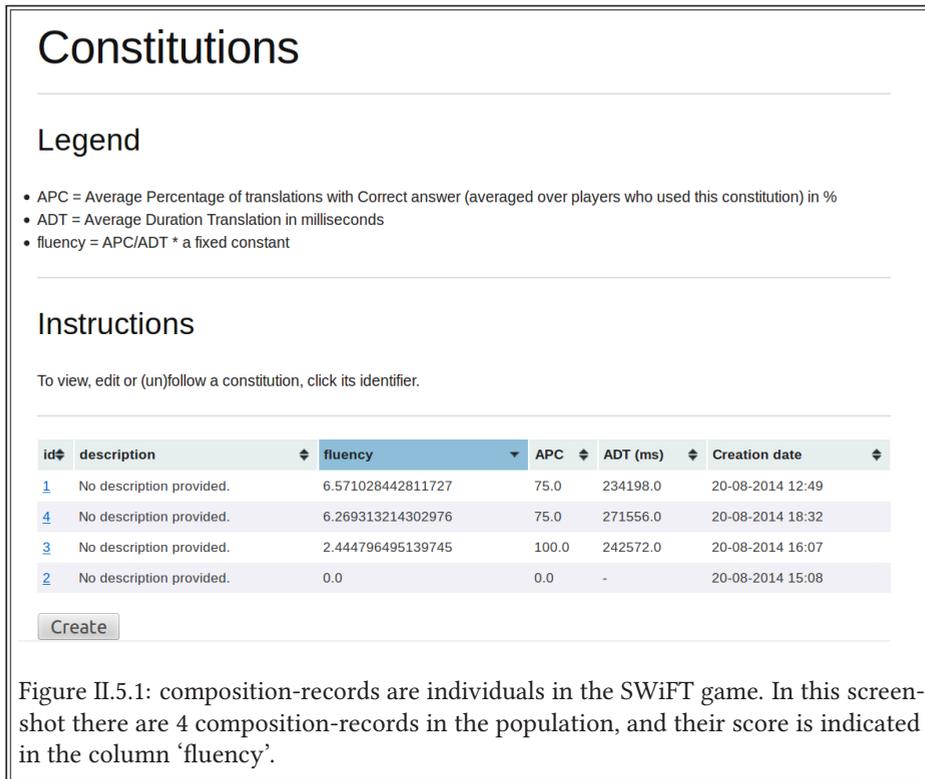
In fig. II.5.2 (p. 80) the individual player scores that make up the sample for the given composition-record is shown in the game environment.

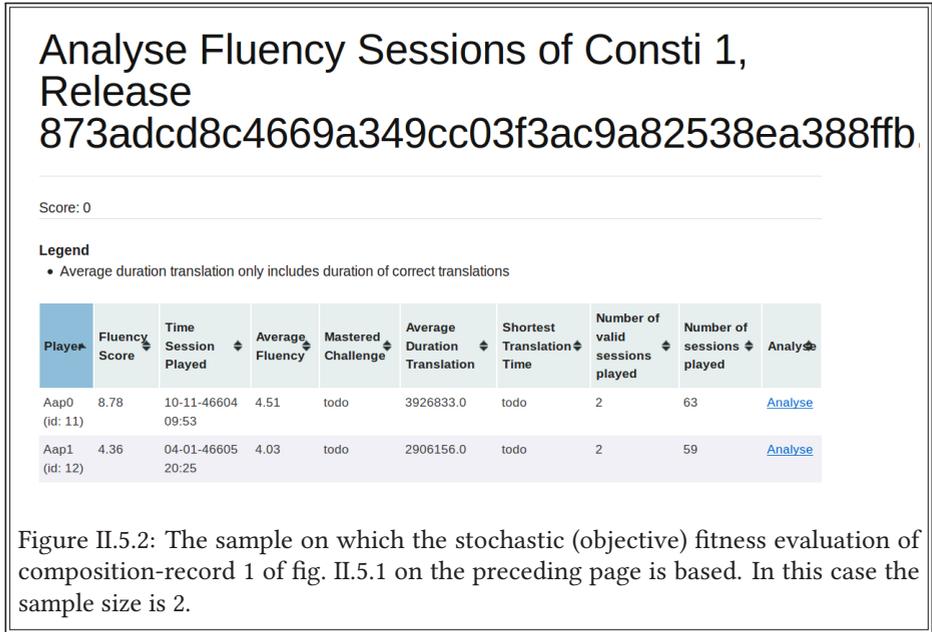
II.5.6.2.2 Objective fitness functions based on dynamic performance

In some Evohuts, the fitness score of evo-individuals is defined in terms of the performance of persons who are interacting with, or who have interacted with that evo-individual.

In an oversimplified world, the performance ‘jumps’ from a level before interaction, to a constant level after interaction, and moreover the interaction is assumed to take no time (or equivalently, one assumes that there are no costs associated with the period of interaction). These assumptions greatly simplify defining a fitness function.

In reality, however, the interaction takes time, and there are economical considerations associated with this: a method that requires shorter interaction with the same performance gains, all other circumstances being identical, is typically considered to be the better method. Second, in reality, the performance does not instantly jump from an initial constant level to a final constant level. In other words, it is not a *step function* as defined in mathematics (Wikipedia contributors, 2019). It is much more





*performance-
function*

dynamical. For example, in many cases the performance will improve as a function of time during the interaction – and after it. Examples of such Evohuts are the previously mentioned basketball team being trained, or the SWiFT game. The term *performance-function* is coined for the latter function. It is defined as follows. Given is an entity that pursues acquiring or improving a certain capability, for example, a person that pursues improving his time on 400 meters freestyle swimming. The performance-function is a function that maps a point in time to the performance-level of this entity. It is assumed that the performance-level is expressed as a real number. A closely related notion is *learning curve*.

For clarity's sake, this subsection assumes that the performance-function can be known with full precision at any moment. Section II.5.6.4 (p. 87) adds uncertainty. The problem is that this dynamic nature makes the definition of a suitable fitness function non-trivial. The dynamic information has somehow to be 'compressed' into a single real number that makes sense as a fitness score for the purposes of the given Evohut.

The subsequent examples aim to illustrate the problem. Suppose that you want to determine the influence of a training method in the aforementioned basketball team. Thus, the training methods are the evo-individuals. Moreover, suppose that the actual

performance-functions for method 1 and 2 are as displayed in fig. II.5.3 (p. 83). Here, we assume that the team is reset to its initial conditions when measuring the effect of each method, a thing which, in reality, of course, never can be done. Moreover, the assumption is that we are interested only in approaching an optimal training-method for that particular team, in contrast to targeting a more general class of teams. One could use the score of the first basketball match the team plays as the score for the method. In that case, method 2 is better than method 1. However, in the same graphs, we see that the stable level reached is higher when using method 1. From that perspective, method 1 is better than method 2. Many more comparisons can be made, for example, one could look at the time to reach the stable level. Consider, for example, a situation in which two methods lead to the same stable level, but that it takes longer to reach it with method 1 than with method 2. Moreover, the form of the performance-function not only depends on the training method, but also on other factors external to it. In the end, it turns out that there is no unambiguous choice that is best: it depends on what you want to achieve with the training method.

The assumption that one is interested only in approaching the optimal training-method for a *particular* team does not hold in many cases. Moreover, it would be impossible to determine the performance-function of other training-methods by exposing these to the same team, because that would require a reset of the team to the initial conditions to try out each method with a 'clean slate'. Time machines have not been invented yet.

The solution is taken from empirical science: subject several teams meeting certain conditions (for example: they should all be female players of junior level) to a given method, and then apply a statistical analysis to make an estimation of the expected performance of a team meeting those conditions. (Note that in this section we still assume that the performance-function of an *individual* team can be measured with complete certainty. In a subsequent section this assumption is dropped as well. This requires the application of additional statistical methods.) In that way, each method's performance can be measured by assigning them to teams that have not been exposed to other training-methods yet, mimicking the 'clean slate'-effect. Moreover, the Evohut can now be used to approach the optimal training-method for any team that meets the mentioned conditions. The objective fitness function is now defined in terms of *several* performance-functions. One way to do this is to first compress the performance-function of each individual team that was trained with the method into one sensible real number (the *performance-index*), and then apply a suitable statistical transformation to all performance-indices of the method to produce the objective fitness value for that method. Note that in the first approach under simplifying assumptions, the performance-index is (equal to) the objective fitness function value, in the latter it is only part of its calculation. shoulddojj what do latter and first refer to.

*performance-
index*

A basic approach would be to take the average of the performance-indices of the teams that played with the method, and requiring a minimal sample size.

II.5.6.2.3 Terminology and assumptions related to performance-functions

The following introduces the scope and some relevant terminology, notions and assumptions. The performance-function is the performance as a function of time. Note, that the notion *learning curve* is closely related.⁵ The scope of this study only includes performance-functions that occur under the condition that training is taking place at regular intervals, which are sufficiently short to prevent a regression in performance, and (2) the conditions, except for the (effect of the) training, remain the same as much as possible. A non-example of the second would be a distraction caused by someone falling in love while training⁶, or, over longer periods of time, a child growing up and simply acquiring new cognitive capabilities as a consequence of the child's physical maturation process. Under these conditions, the actual performance-function is monotonically increasing. Moreover, this study assumes it is constant at and from a certain moment, i.e. from this moment, no further progress is made. The term *flatline-moment* is coined for this moment. The *flatline-level* is the value of the constant, which is equal to the maximum of the function. Whether the aforementioned scope is adequate and the assumptions are valid, or at least a reasonable approximation of reality, has yet to be determined. For example, a better representation of the flat-line level may be an asymptote, so a line that will never be reached. For our purposes, however, the flatline model is sufficiently close.

learning curve

flatline-moment
flatline-level

II.5.6.3 Performance-index candidates

The fact that the actual performance-function is not constant makes it more complicated to define a suitable objective fitness function, as pointed out in section II.5.6.2.2 (p. 78). For this reason, the notion 'performance-index' was introduced. These indices compress a performance-function into a single number. The following presents several proposals for performance-indices. Which index is better than another may depend on the characteristics of a particular Evohut.

For all clarity: the reader may be tempted to confuse the notions 'performance-function' and 'fitness function'. In fact, however, the former is *defined* in terms of the latter, and different ways to do this are presented in the following. Moreover, as

⁵Note that the notion 'a steep learning' curve is in fact used incorrectly: it would mean that learning progresses quickly in an early stage instead of what one intends to say: the opposite.

⁶It is left to the reader's undoubtedly rich fantasy whether this will lead to a better, or worse performance.

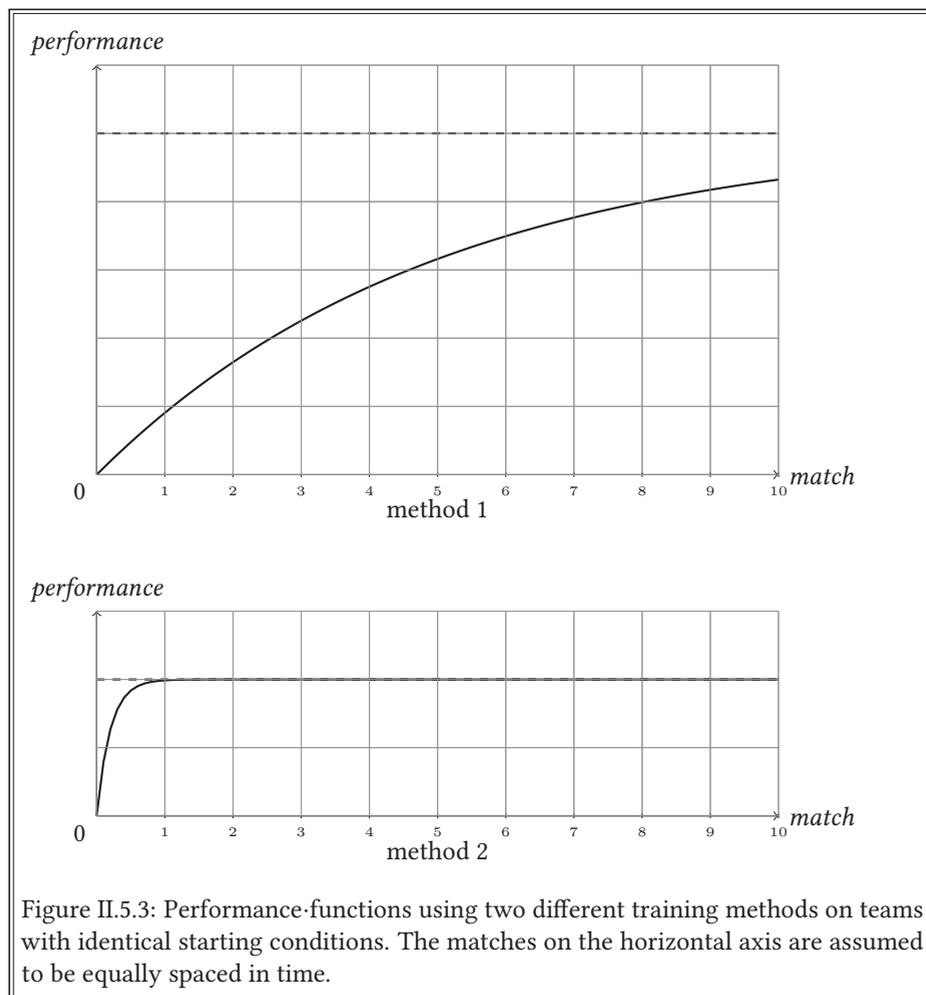


Figure II.5.3: Performance-functions using two different training methods on teams with identical starting conditions. The matches on the horizontal axis are assumed to be equally spaced in time.

stated before, the words ‘fitness score’ and ‘score’ are used synonymously, and mean ‘the outcome of a fitness evaluation’.

II.5.6.3.1 Notation

The performance-function of composition-record-player pla is formally expressed as follows:

$$\text{perf}_{pla}. \quad (\text{II.5.6})$$

A performance-index function of a given performance-function pf is expressed using the following notation:

$$\text{perf_index}_{label}(pf) = [\text{definition}], \quad (\text{II.5.7})$$

where $label$ is a unique label to discern different performance-indices from each other. The flatline-level of a performance-function pf is indicated as follows:

$$\text{flat_lev}(pf), \quad (\text{II.5.8})$$

and the flatline-moment as follows:

$$\text{flat_mom}(pf). \quad (\text{II.5.9})$$

II.5.6.3.2 The ‘traditional’ approach: performance-index at a fixed point of time $t = [\text{some constant}]$

The performance-index defined by the performance level at a fixed time t_c ,

$$\text{perf_index}_{trad}(pf) = pf(t_c), \quad (\text{II.5.10})$$

is reminiscent of the ‘traditional’ approach that is often used in settings where human capabilities are trained and evaluated, such as in most schools and universities. People are trained for a fixed amount of time, and right after that an examination takes place. This index may be useful if the amount of training time is fixed in advance. A great disadvantage, among other things, is that this index is unable to distinguish between a performance function reaching its flatline-level in a time that is shorter than t_c , and a function in which reaching the same flatline-level takes exactly t_c time. Therefore, it does not incentivise the method developers to increase the performance gains in a shorter time than the predefined time span, or, to extend the training time because people are still making progress. (Note that in traditional educational settings the final evaluation (exam) is not the only evaluation. There are, in most cases, many informal performance measurements during the training, for example in the form of a professor P talking to students while they making exercises. Based on P ’s experience, P may adapt the training method *on the fly*. The underlying assumption in this section, however, is that the method is fixed.)

II.5.6.3.3 The flatline level (perf_index_{fl})

The performance-index defined by the flatline-level of the function pf ,

$$\text{perf_index}_{fl}(pf) = \text{flat_lev}(pf), \quad (\text{II.5.11})$$

may be useful if the influence of the evo-individual is expected to have a dominating influence on the flatline-level. *Example: Consider an Evohut in which the evo-individuals are formed by piano pedagogues. It is a well-established fact that, in general, these have a tremendous influence on the final performance-level of their pupils.* This approach will not be suitable for contexts in which influencing the flatline-level falls outside of the scope of the Evohut, such as the computer-manual example mentioned in section II.5.6.3.4.

II.5.6.3.4 The (inverse of the) flatline-moment (perf_index_{fm})

The performance-index defined by the inverse of the flatline-moment of a given performance-function pf ,

$$\text{perf_index}_{fm}(pf) = \frac{1}{\text{flat_mom}(pf)}, \quad (\text{II.5.12})$$

may be useful, if it is to be expected that the evo-individual does not influence the flatline-level, but does influence the flatline-moment. Another way to phrase it, is that the score is the time it takes to realise the maximum learning potential (given the method). *Example: Consider an Evohut in which the evo-individuals are manuals for how to operate the on/off switch of a certain model of computer, and the performance measured is the time it takes a person to turn on a computer of the given model. As soon as the person has figured out how to turn it on, the speed will not any longer depend on the manual (the evo-individual), but mostly on the motor skills and focus of the person.*

II.5.6.3.5 The surface from $t = 0$ and $t = [\text{some constant}]$ (perf_index_{stc})

The performance-index of a given performance-function pf defined by the surface under the graph from $t = 0$ to a given predefined point t_p , is formally expressed as follows:

$$\text{perf_index}_{stc}(pf) = \int_0^{t_p} pf(t)dt, \quad (\text{II.5.13})$$

where stc is a label that stands for ‘surface to constant’. The index can be normalised by dividing it by t_p , leading to an index that is equivalent. (This form is needed to

show some relations with other scores later on.)

$$\text{perf_index}_{stcn}(pf) = \frac{\int_0^{t_p} pf(t)dt}{t_p}, \quad (\text{II.5.14})$$

where *stcn* is a label that stands for ‘*surface to constant – normalised*’. It is motivated as follows. If the range of the performance-function expresses a *speed* of something, then the surface under the graph represents the amount of work the composition-record-player could have done applying the capability. If the capability is for example translation from natural language to a formal-language, then that would be the amount of translation work the composition-record-player could have done.

Some relations with other performance-functions are as follows. The flatline-level as score (section II.5.6.3.3 on the previous page) is identical to taking the limit of t_p to infinity for the normalised score (eq. (II.5.14)), by choosing the right value for constant c .

A disadvantage of this function may be that it is in a sense underspecified: it contains a parameter t_p , which still has to be determined ad hoc. An advantage is that the parameter t_p , given knowledge about the purpose of the Evohut, may be exactly the parameter you want to tweak.

II.5.6.3.6 The surface from $t = 0$ to

$t = [\text{the latest flatline-moment}]$ ($\text{perf_index}_{stlfm}$)

Given a set of performance-functions, then the performance-index of a performance-function pf based on the surface to the latest flatline-moment is defined as follows. Take the surface under the plot from $t = 0$ to the latest flatline-moment that occurs in the set of performance-functions. The surface is the performance-index of the composition-record-player with the given performance-function. Formally expressed, this becomes:

$$\text{perf_index}_{stlfm}(pf) = \int_0^{t_{laflamo}} pf(t)dt, \quad (\text{II.5.15})$$

where:

- The label ‘stlfm’ stands for ‘Surface To Latest Flatline Moment’.
- $t_{laflamo}$ is the latest flatline-moment that occurs in the set of performance-functions.

Figure II.5.4 (p. 88) illustrates this: in this case the set consists of 3 actual performance-functions. The function in the middle has the latest flatline-moment, so this is set as the reference for the whole set. The surface under the lowest plot is highest, so the

performance is highest, while the plot on the top follows as the second best of this set.

Note that this is a relative measure, one can only determine the score with respect to other performance-functions (because the latest flatline-moment depends on the functions compared!).

The choice for taking the surface to *the last flatline-moment in the set* is motivated by the following. First, an earlier moment would not integrate all ‘dynamic’ information in the score, i.e. not each point of each graph in the set that has not reached its flatline-level is part of the calculation. Second, a later moment would not maximise the effect of the period wherein there is still progress in performance (note that if you take the limit of time to infinity, this effect becomes zero). Another property of this performance-index is that a slower progress, but a higher final performance-level, can be compensated with quick progress, but a lower final performance-level. Whether this is desirable, depends on the purpose of the capability: is it intended to be put into use over a long time, or not? If it is intended to be used over a very long time, the advantage of the quicker progress becomes negligible over time.

II.5.6.3.7 The surface between $t = 0$ and

$$t = [\text{some constant}] \times [\text{the latest flatline moment}] (\text{perf_index}_{stclfm})$$

The performance-function $\text{perf_index}_{stclfm}$ is almost the same as the one in section II.5.6.3.6 on the facing page, except for choosing as distance the latest flatline-moment multiplied with some constant $c > 1$. The advantage is that this will favour higher flatline-levels (in comparison to lower flatline-levels in a shorter time). Its definition is:

$$\text{perf_index}_{stclfm}(pf) = \int_0^{c \cdot t_{laflamo}} pf(t) dt, \tag{II.5.16}$$

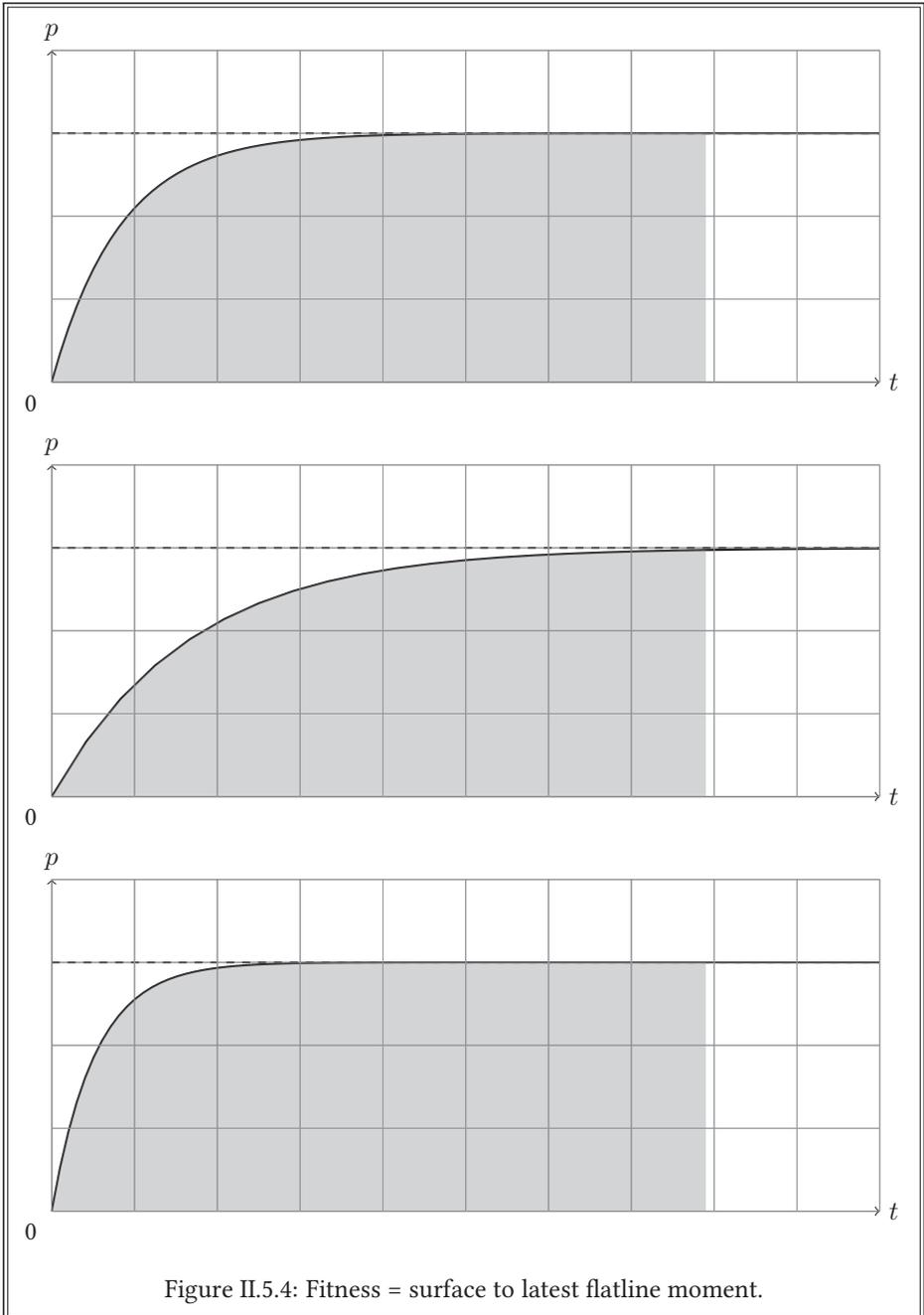
where

- The label ‘stclfm’ stands for ‘Surface To Constant times Latest Flatline Moment’.
- $t_{laflamo}$ is the latest flatline-moment that occurs in the set of performance-functions.
- c is some constant.

II.5.6.4 Stochastically measured dynamic performance

The actual performance-function can never be known with complete certainty, because of uncertainties in measurements. To be precise: each measurement of the performance-level at a specific point in time (the *performance-measurement*) is, to

performance-measurement



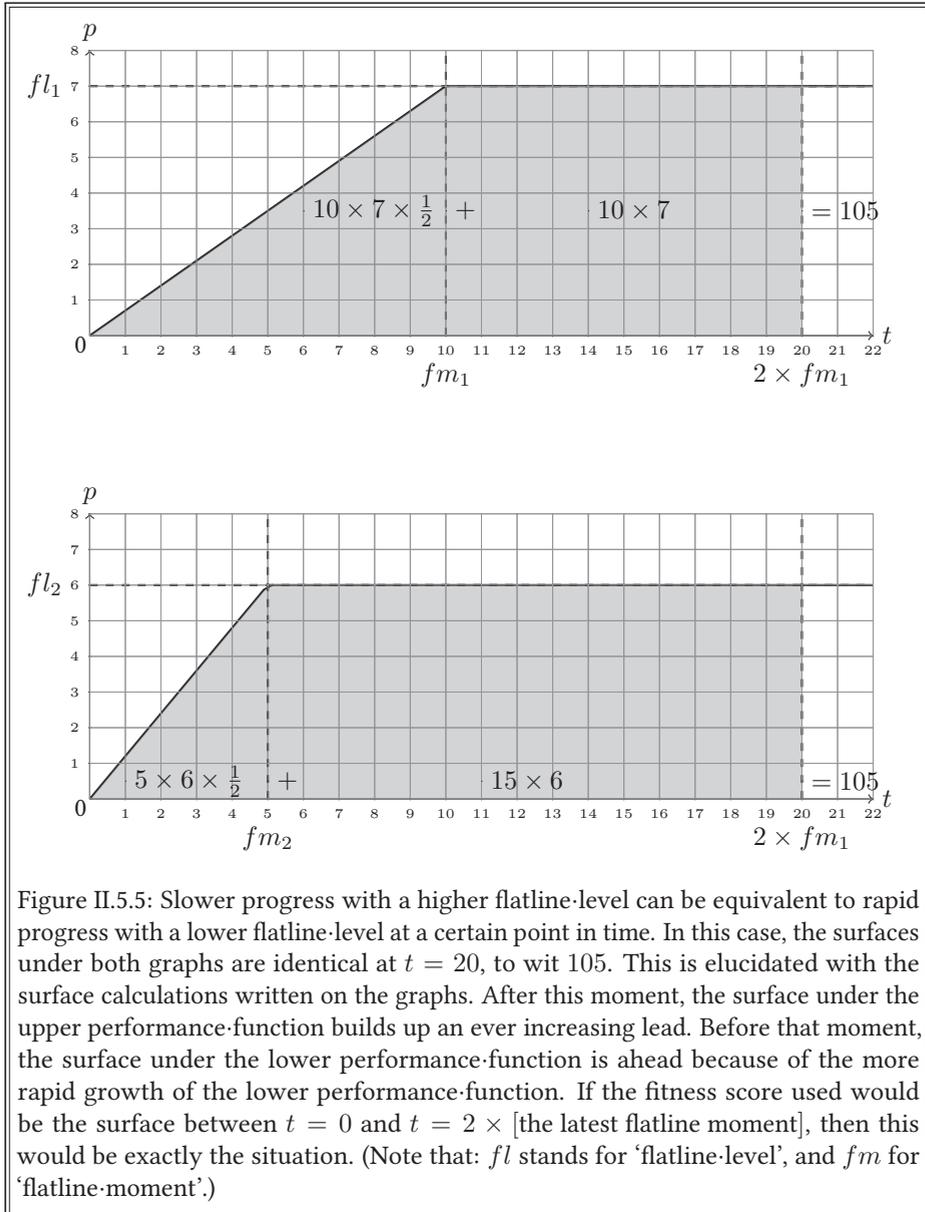
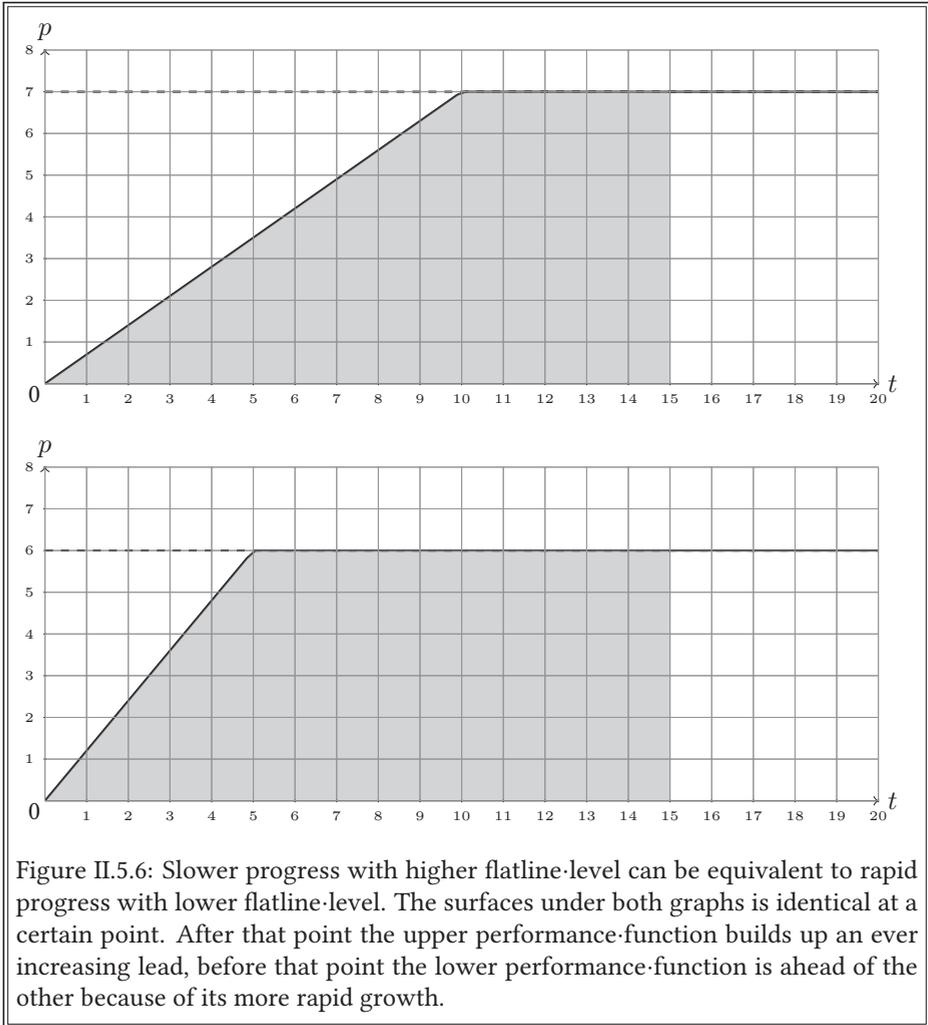


Figure II.5.5: Slower progress with a higher flatline-level can be equivalent to rapid progress with a lower flatline-level at a certain point in time. In this case, the surfaces under both graphs are identical at $t = 20$, to wit 105. This is elucidated with the surface calculations written on the graphs. After this moment, the surface under the upper performance-function builds up an ever increasing lead. Before that moment, the surface under the lower performance-function is ahead because of the more rapid growth of the lower performance-function. If the fitness score used would be the surface between $t = 0$ and $t = 2 \times$ [the latest flatline moment], then this would be exactly the situation. (Note that: fl stands for 'flatline-level', and fm for 'flatline-moment'.)



some degree, uncertain. (Related statistical terms are *measurement precision* and *observational* or *measurement error*.) This book uses the term *performance-sample-data* to indicate the set of these performance-measurements. The performance-sample-datas forms a measurement of the performance-function (the *performance-function-measurement*). For example, if one would like to assess the performance of a human text translator *Trans* from English to Igbo, one could offer *Trans* a random text with a fixed size and complexity to translate, and measure the time *Trans* needs to translate it flawlessly. However, it is impossible to determine (or even define!) the *exact* complexity of a text. Moreover, there are always fluctuating external factors, such as distractions and so forth.

The performance-function, however, can be *estimated* based on these measurements. The measurements are taken at discrete intervals of time, therefore, it is an example of a so-called *discrete-time series*. The art of estimating and predicting time series as good as possible, has professionalised in the form of *time series analysis*. For example, if the measurements have a degree of uncertainty, estimation is desirable.⁷ Section II.5.6.5 provides a brief elaboration on relevant theories from statistics, including time series analysis, that are to be applied in the sections to come. For the convenience of the reader, it also introduces some common basic probability theory notions and definitions that forms their foundation.

II.5.6.5 Relevant statistical theory

This section introduces relevant statistical theory.

II.5.6.5.1 Basic definitions of probability theory

Modern probability theory revolves around the notion of the *random experiment*, which indicates an experiment that has an uncertain outcome, such as throwing dice, tossing a coin, or measuring the power output of a solar panel. Common probability theory adopted Kolmogorov's formalisation. In the latter, the random experiment is formalised as a so-called *probability space*, which consists of the following elements.

1. A set of possible outcomes, the sample space Ω .
2. A set of *events* \mathcal{F} . Each event consists of a set of outcomes from Ω . An event has *happened* if the outcome of the random experiment is a member of the event. The set of events, moreover, includes the empty subset, is closed under complement, and is closed under countable unions and countable intersections (which makes it a σ -algebra).

⁷It is more generally applicable to a larger class functions of variables that are associated with time.

probability
measure

3. A *probability measure* P which assigns a probability to each event. Each probability takes a value in the interval $[0, 1]$. The P of the union of a countable number of mutually disjoint events, should be equal to the sum of the P of each of these events (countable additivity). The ‘semantical’ assumption behind this definition is that if the random experiment would be repeated an infinite number of times, the relative frequencies of occurrence of each of the events would coincide with the probabilities prescribed by P .⁸

A probability space is written as a triple (Ω, \mathcal{F}, P) .

Further basic concepts that build upon this foundation include the random variable, the probability distribution, and the stochastic process.

random variable

A *random variable* is a variable quantity whose value depends on the outcome of a random experiment. For example, if the random experiment is a coin toss, the value could be ‘1’ if the result is heads, or ‘0’ if it is tails. In other words, a random variable is a function that maps the sample space to numbers, for example to \mathbb{R} (real numbers) or \mathbb{N} (natural numbers). An additional property that has to hold is that this function is measurable, of which the explanation is beyond the scope of this book.

stochastic process

Instead of a single random variable, often one wants to investigate a set of random variables that are related to each other. Examples are the performance of an athlete over the course of several months, the power output of a solar panel during a given hour. The *stochastic process* is the formalisation of such a set. Typically, its random variables are indexed by a set of numbers. Usually these numbers represent points in time, just as in the examples just given. Formally, a stochastic process is defined as a collection of random variables defined on a common probability space (Ω, \mathcal{F}, P) . The random variables are indexed by some set T . They all take values in the same mathematical space S .

II.5.6.5.2 Statistical estimation theory and sequential analysis

The basic problem of statistical estimation theory is estimating a statistical parameter of a population (Lehmann and Casella, 2006). (Population parameter is a synonym of statistical parameter.) The basic approach is repeatedly drawing a random individual from that population and observing the outcome. The resulting collection of relevant values associated with each outcome is called the random sample. The random sample is statistically analysed to produce an estimation. An example is the estimation of the average height of Dutch people, based on a measuring the height of a random selection Dutch people.

⁸This is the so-called *frequentist* interpretation. Alternative interpretations, such as the Bayesian interpretation, exist.

A common formalisation of the aforementioned problem is as follows. The problem is estimating an unknown statistical parameter θ from the probability distribution of a random variable X . X fulfills the same role as the aforementioned population. In a sense, it is an abstract representation of that population. The sample is created by repeatedly carrying out X . Hence, it consists of a sequence of random variables $\{X_0, \dots, X_{n-1}\}$ with the same distribution as X . An observation of X is then used to estimate the unknown θ .

The following definition summarises, and adds to the terminology introduced so far:

Definition 11 (notations estimation theory).

X : the random variable of which one wants to estimate a parameter.

θ : the estimated parameter.

$\{X_0, \dots, X_{n-1}\}$ (or briefly X_*): a sample drawn from X , in other words, repeating X n times.

$\{x_0, \dots, x_{n-1}\}$ (or briefly x_*): an observation of X_* . ■

Note that the previous deviates slightly from the often applied, but somewhat confusing notation of the observed sample with x instead of x_* . In the first case, one may confuse x with being an observation of X , while it is an observation of X_* , so $\{X_0, \dots, X_{n-1}\}$.⁹

II.5.6.5.3 Sequential analysis

There are several different approaches to statistical estimation theory. One of these is sequential analysis (Wald, 1947).

Sequential analysis is a statistical technique, which to the best of my knowledge, *sequential analysis* was first developed by Wald (1947).¹⁰ It dynamically determines the sample size by defining a desired certainty level. Collecting new measurements is stopped as soon as the desired certainty level is reached. The great advantage is that, in general, it requires fewer measurements (a smaller sample size), than the standard approach with a predefined, fixed, sample size, and that one prevents the risk of ending up with a sample with a precision that is too low

The following presents the standard *sequential procedure* (Wald, 1947). It is important to be aware that this sequential procedure cannot be adopted 'as is'. The *sequential procedure*

⁹I encountered deviating notations in the literature. For example, one textbook defined $X = \{X_0, \dots, X_{n-1}\}$ instead. In that case, using the notation x to indicate $\{x_0, \dots, x_{n-1}\}$ is consistent. Another textbook, however, used X in the way adopted in this book.

¹⁰I 'rediscovered' this theory before I was aware that is an established field of statistics, thanks to statistician Kruijer (2013).

procedure is a ‘parameterised procedure’. One needs to customise it to one’s specific use case by filling in some ‘blanks’. How this is done, depends, among other things, on the demands from the given use-case.

First, sequential analysis needs the definition of a function that selects a confidence interval with a given confidence level (or better), based on a given observation x_* .

This function is not uniquely determined by the confidence level, so the following merely describes the basic condition it should meet.

Definition 12 (Condition confidence interval function).

$$P(\theta \in \text{conf_int}(x_*)) \geq \gamma, \quad (\text{II.5.17})$$

■

Ideally, `conf_int` should be chosen such that the chosen confidence interval is smallest given the observations. In other words: one seeks the most precise estimation possible given the available observations. Which one is smallest, however, is generally not decidable, and even then it is (probably) not even uniquely determined. Hence, instead, one tries to approach the smallest choice as good as possible.

The procedure furthermore requires a way to express which confidence intervals produced by `conf_int` are to be considered to be acceptable, given an observed sample x_* .

Definition 13 (acceptance set). *There is a function `accept_set` : $2^{\Omega^n} \rightarrow \text{Bool}$, where Ω is the sample space of X .¹¹ In words, this acceptance set maps a sample to a truth-value.*

■

Note that the function `accept_set` corresponds with the first condition in Wald (1947, Sec. 11.2). Its further customisation is not a statistical problem, but determined ad-hoc based on the specific estimation problem and the desired level of precision. For example, if one wants a very precise estimation, one has to define it such that the intervals it accepts are very narrow. It is the intention of the formalisation of sequential analysis that I presented here to define the function `accept_set` in terms of the `conf_int`.

It may be somewhat confusing that whatever one chooses as the function `accept_set`, it can never violate definition 12, not even indirectly. The problem lies somewhere else. If one chooses it inadequately, it may accept some samples that have a confidence interval that is wider than one desires. In the worst case, the interval may even cover the entire domain of the estimated parameter, which means that the estimation does not produce any valuable information.

¹¹ 2^S is the standard notation for the power set (Wikipedia, 2018a).

II.5.6.5.4 Customisation of the sequential procedure

This section presents a possible customisation of the sequential procedure to arrive at one that is fully specified. (The choice for the presented customisation is not arbitrary. It is applied in later sections.)

A natural choice for the confidence interval is one that is ‘balanced’ around the mean of the sample. In other words, choose the confidence interval such that the mean of the sample is in the middle of the interval. Self-evidently, the width of the interval should be such that it complies with definition 12 on the preceding page, and is as narrow as possible.

Formally one can deduce the following (fully specified) definition for `conf_int` given these additional constraints. For this, one needs *Student’s t-distribution* (Cramér, 2016):

Student’s t-distribution

Notation 1 (Student’s t-distribution). *Let f_ν be Student’s t-distribution with ν degrees of freedom. Let F_ν be the cumulative Student’s t-distribution with ν degrees of freedom, and F_ν^{-1} the inverse function of F_ν .*

This book assumes that F_ν^{-1} is a known function. In other words, the reader is able to evaluate it in any point. ■

The following assumes that X has a normal distribution with an unknown variance $Var(X)$ and an unknown mean $E(X)$. Then:

$$T = \frac{\bar{X}_* - E(X_*)}{Var(X_*)/\sqrt{|X_*|}}, \tag{II.5.18}$$

has Student’s t-distribution with $\nu = |X_*| - 1$ degrees of freedom. The central question now is, for which c the following holds:

$$P(-c < T < c) = \gamma$$

This is equivalent to the statement:

$$\int_{-c}^c f_\nu(t) dt = \gamma$$

Considering that the total surface under f_ν is 1, and that it is symmetrical around $t = 0$, it is easy to see that:

$$\int_{-\infty}^c f_\nu(t) dt = 1 - \frac{1}{2}(1 - \gamma) = \frac{1}{2} + \frac{1}{2}\gamma.$$

By notation 1 on the preceding page:

$$F_\nu(c) = \int_{-\infty}^c f_\nu(t) dt,$$

so,

$$F_\nu(c) = \frac{1}{2} + \frac{1}{2}\gamma$$

Consequently,

$$c = F_\nu^{-1}\left(\frac{1}{2} + \frac{1}{2}\gamma\right)$$

All that is left to do, is a simple transformation of eq. (II.5.18) on the previous page so that it expresses the probability in terms of X_* instead of T :

$$P(-c < T < c) = P\left(\bar{X}_* - \frac{c \cdot \text{Var}(X_*)}{\sqrt{|X_*|}} < X_* < \bar{X}_* + \frac{c \cdot \text{Var}(X_*)}{\sqrt{|X_*|}}\right) = \gamma.$$

This provides exactly the information needed to define the confidence interval function, which takes as input an observation of X_* (as indicated earlier, written as x_*):

Definition 14 (balanced confidence interval).

$$\text{conf_int}_{\text{balanced}}(x_*, \gamma) = \left[\bar{x}_* - \frac{c \cdot \text{Var}(x_*)}{\sqrt{|x_*|}}, \bar{x}_* + \frac{c \cdot \text{Var}(x_*)}{\sqrt{|x_*|}} \right], \quad (\text{II.5.19})$$

where

$$c = F_\nu^{-1}\left(\frac{1}{2} + \frac{1}{2}\gamma\right)$$

■

I believe that a suitable and sensible choice for an acceptance set based on this `conf_int` is acceptance-by-percentage-deviation-from-mean defined as follows.

Definition 15 (acceptance-by-percentage-deviation-from-mean).

$$\left(x \in \text{accept_set_perc_dev}\right) \Leftrightarrow \left(\frac{\frac{1}{2}\text{conf_int}_{\text{balanced}}(x, \gamma)}{\bar{x}} \leq \delta\right),$$

where δ is a constant that has to be chosen by the person who customises the sequential procedure, and \bar{x} is the sample mean of x . δ expresses the desired maximum percentage-wise deviation from the mean. Hence, $\delta \in [0, 1]$. ■

A sensible choice, depending on the context, may for example be $\delta = 0.10$. This means that the confidence level is at least γ that the actual value of the statistical parameter is within 10% of the mean of the sample.

II.5.6.5.5 Regression analysis

Regression analysis is about estimating the relation between several variables in the face of uncertainty. Typically, these variables have a numeric value, and the relation to be estimated can be expressed as a mathematical function, the *regression function* (Sen and Srivastava, 2012). In other words, one wants to express how a set of given variables depend value-wise on a set of other variables. Typically, one focuses on sets that contain exactly one variable, for example: how does the variable temperature depend on the variable pressure. In this context, the first set of variables, that serve as the input of the function are called the independent variables. The second set, that form the output of the function are called the dependent variables. *regression function*

Examples are the relation between the solar power (brightness of the sun) and the power-output of a solar panel, the heart rate of person as a function of time, the weekly sales of solar panels as a function of their price, the response of a person to a medicine as a function of the person's blood pressure, and the value of a student's performance as a function of the hours of training.

Major sources of uncertainty include uncertainty in measurement or unavailable data. An example of uncertainty in measurement is in the previous example of the student. Typically there are no perfect ways to measure a student's true performance for a given capability. An example of unavailable data is the previous example of the response of a person to a medicine – typically such a response cannot be measured continuously, but is limited to points in time, for example once a day. The actual situation between measurements can only be estimated.

Regression analysis is one of the most common research instruments in the repertoire of empirical research, ranging from clinical research to astrophysics. Therefore, one may be inclined to think that (1) methods for regression analysis would have matured beyond the need of the development of additional or better methods, and (2) that a few general methods for regression analysis would suffice¹², and (3) that terminology used within the field has been completely standardised (*end of list*).

Nothing, however, is farther from the truth. Yearly, new research papers are published describing new or refined methods for regression analysis and associated techniques. There are many approaches to regression analysis, some being mutually distinct, others being related to, complementary to or refinements of other methods. The reasons for this include: (1) The quality or applicability of a method depends on the model assumptions made, and the latter depends on the domain of discourse. Given the fact there are an ever-growing number of (sub)domains, it is also to be expected that an ever-growing repertoire of methods is desirable. (2) The best method to apply, depends on the quality and quantity of the collected data. For example, the

¹²These were assumptions that were originally, perhaps subconsciously, held by the author. However, they proved to be incorrect, leading to the inclusion of these paragraphs.

method that performs well on a sparse data set may perform badly on a dense data set. (3) Various trade-offs, including computational complexity versus accuracy of the estimation. For example, some methods have a higher accuracy, but require an unreasonable amount of computer resources. (*End of list.*)

Added to this is the fact that there are different sub-communities that are involved in application and development of regression analysis and associated statistical techniques. These communities sometimes employ different names for similar or even identical techniques. For example, the term ‘panel data’, used within econometrics, is equal to the term ‘longitudinal data’ used among empirical researchers. This makes it harder to get an overview of suitable methods for analysing a given set of data. Even the name regression analysis itself seems to be used with slightly different meanings.

Moreover, some communities may emphasise other aspects within regression analysis. For example, in economy and meteorology, forecasting is emphasised. Forecasting is estimation of the regression function, where the independent variable is time, and at locations where no data points about the function have been collected yet. For example, given the measurements made up to this moment, what is the value of a stock, or the temperature the next day? On the other hands, a large part of empirical research, concerns itself with estimations of the true value of a variable as function of time *between* or *at* the times of the data points. (Estimation at the data points may be needed because there is often uncertainty in measurement). For example, given the measurements made in the past year on a set of test persons undergoing a certain treatment, estimate a regression function over the past year. This is probably the reason that ‘time series analysis’, a term used predominantly within the domain of econometrics, often leads to literature about forecasting and prediction, that is statistical inferences *after* the measured data points, while ‘longitudinal data analysis’ predominantly to techniques for statistical inferences *between and at* the measured data points. The latter is mostly used for finding out how something behaved in *the past* (such as the interaction between a pharmaceutical and a person).

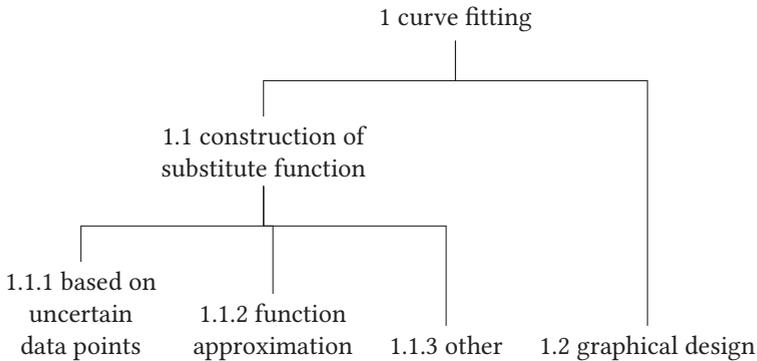
This is associated with what I will call ‘overgeneralised terminology’. I define it as the tendency of research communities to coin a term that has a meaning that is too general for the concepts and activities that are, often implicitly, considered to fall within the scope of that term. An example is the notion time series analysis which is, purely interpreted, the statistical analysis of data resulting from measurements of a variable in time. However, the term is mostly used within the fields such as econometrics, in which certain quite specific techniques are tacitly associated with the notion, while others are not. For example, in Wikipedia (2018e), the relation between regression analysis and time series analysis is considered to be different than with a ‘pure’ interpretation of the terms. In the pure interpretation, all forms of statistical analysis of time series should be considered to be part of time series analysis. However, for example, using time series as part of a regression analysis

of a given dependent time series is *not* considered to be part of time series analysis according to Wikipedia (2018e).

The same, holds for the notion regression analysis itself – some interpret this as a bundle of *specific* techniques to statistically estimate functions, which, for example includes the method of the least squares, but not certain techniques used in time series analysis .

II.5.6.6 Curve fitting and regression analysis

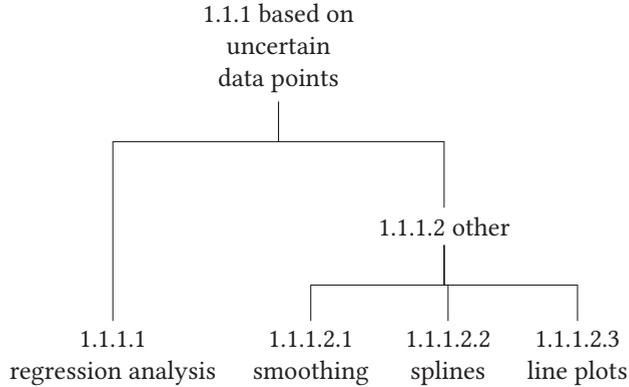
Another term that is highly related to regression analysis is curve fitting. *Curve fitting* is the process of constructing a curve, or mathematical function, that fits a series of data points (Arlinghaus, 1994). There does not seem to be a clear consensus within the scientific community about the differences between the two terms. An interpretation is that curve fitting is more general than regression analysis, in the way elucidated in the following diagrams.



The diagram shows, among other things, that a main application area of curve fitting consists of the construction of a function that acts as a ‘substitute’ for the ‘true’ pursued function, that somehow cannot be accessed directly (see 1.1 in the diagram). Note that the previous and following diagrams also include some curve fitting topics that are not relevant to this work, such as graphical design. They are included because they may help finding one’s way through the jungle of statistical and related methods if one wants to extend this work. The abundance of terminology can be quite daunting and confusing.

A main subcategory of curve fitting for constructing a substitute function (1.1) is the situation in which only data points with uncertainty are present, for example when the determination of the function has to be based on measurement that intro-

duces random errors (1.1.1). This is the central problem in regression analysis, as indicated in the following diagram (which is a continuation of the previous diagram):



However, the diagram also indicates that curve fitting based on uncertain data points (1.1.1) is not synonymous to regression analysis (1.1.1.1). There are more approaches to this, albeit less precise and rigorous (1.1.1.2). Regression analysis discerns itself in tackling this problem in that the estimation of the function is based on minimising statistically estimated errors in a mathematically rigorous way. Other approaches, for example based on splines (1.1.1.2.2), or the line plots (1.1.1.2.3) do not necessarily do this. A *spline* is a special function defined piecewise by polynomials (Schoenberg, 1946; Ahlberg et al., 1967). A *line plot* is one of the simplest forms of curve fitting, and consists simply of connecting all adjacent data points with a straight line. *Smoothing* is the creation of an approximating function that attempts to capture important patterns in the data, while leaving out noise or other fine-scale structures phenomena (1.1.1.2.1). A general approach is to move data points that are in each others vicinity closer together in the vertical direction, to allow a ‘smoother’ curve to run through them. A well-known technique that works this way is the moving average and many variants of it. Smoothing is yet another sub-field that is not sharply defined.

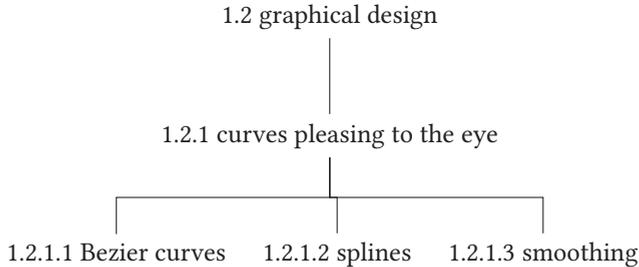
spline
line plot
smoothing

Note that there exist all kinds of combinations of the aforementioned techniques. For example, there are extensions of the spline technique, such as regression splines (Friedman, 1991).

The non-regression analysis techniques are useful for the purpose of curve fitting through uncertain data points, for example if the time to create a precise statistical model is limited, or when information about the domain is so limited that it is difficult to create a trustworthy statistical model.

For completeness, I want to note that curve fitting can also be applied to a completely other type of problem than pursuing a substitute for an inaccessible

function. An example is the application of curve fitting in the design of computer graphics as indicated in the following diagram.



In this case curve fitting is, for example, simply intended to create a curve that is pleasing to the eye (1.2.1), such as in the design of a logo for an organisation. The same sub-techniques as used within curve fitting (such as splines and smoothing) can be applied.

Non-linear regression analysis can be used to fit a function from a given class of functions to the measured data. For this one needs some quality-of-fit metric. A typical choice is the least squares metric.

A disadvantage of this ‘ordinary’ regression in comparison with time-series analysis, is that time series form a richer model. In time series it is possible to define and exploit dependencies between the random variables of which it is composed. (Let the reader be reminded that time series is a set of random variables, for each point of time one.)

II.5.6.6.1 Time series analysis

Typically, time series analysis is about analysing observations of a certain quantifiable property of some entity in time. Examples of such observations are the power output of a solar panel, the value of the share of a company, or the heart rate of a person, and indeed, the performance-function. The purposes of time series analysis include predicting or estimating the given property as a function of time. The art of estimating and predicting time series as good as possible, has professionalised in the form of *time series analysis*. For example, if the measurements have a degree of uncertainty, estimation is desirable.¹³

*time series ana-
lysis*
time series

A *time series* is commonly formalised as an observation of a stochastic process.¹⁴ As the name ‘*time series*’ suggests, the stochastic process is the sub-category of

¹³It is more generally applicable to a larger class functions of variables that are associated with time.

¹⁴There are *other* ways of modelling time series, for example as a regression model in which the observed property is regarded as a dependent variable on the independent variable time. Moreover, based

stochastic processes in which the random variables are indexed by a number that represents *time*. Moreover, time series normally consist of observations that are taken at discrete intervals in time. Therefore, time series are most commonly formalised as an observation of a discrete-time stochastic process. Formally expressed, this means that the set of indices T is isomorphic to \mathbb{N} . Typically, \mathbb{N} is chosen for T . Note that there seems to be some contradicting terminology in the literature. According to some sources, time series can also be continuous. In that case, one has to add an adjective to disambiguate: ‘discrete time series’, or ‘continuous time series’. However, in this book we assume that time series are always discrete. Moreover, the reader may from now on assume that ‘time series’ are always observations of a ‘discrete-time stochastic process’.

Note the difference between ‘an observation of a stochastic process’ and ‘a stochastic process’. A time series is the first. The relation between the two is identical to that between a random variable X and an observation x of that variable, in other words, x is an outcome of carrying out the random experiment associated with X exactly once. Equivalently, a time series is the result of actually carrying out the random experiments associated with the stochastic process. To be precise, it is constructed by observing each of the random variables in the given stochastic process and collecting the results.

A common class of models used in time series analysis assume that each random variable of the discrete-time stochastic process depends linearly on previous random variables.

II.5.6.6.2 Longitudinal data analysis

*longitudinal
data analysis*

Longitudinal data analysis occupies itself with the statistical analysis of longitudinal data, that is, measurements of the same individual taken repeatedly in time (Fitzmaurice et al., 2008, p. 2). Repeated measures is sometimes used as a synonym for longitudinal data. However, according to Fitzmaurice et al. (2008, p. 425) it is not pure synonym, but a subclass or a related class of longitudinal data. It is important to note that a prevalent, but incorrect, definition of longitudinal data analysis circulates: one that assumes that the time intervals between measurements are quite large, ranging from days to even years. However, in longitudinal data analysis the only requirement is that the time intervals are sufficiently large for any meaningful change to have taken place. This implies that these intervals can be of any size, ranging, for example,

on the usage of the term time series analysis I encountered in the literature it seems that many *identify* the field with this model. The name of the field, seems therefore somewhat confusingly to be used in a more restricted sense than its name suggests.

from seconds, to decades.¹⁵

II.5.6.7 Statistical estimation of the performance-function

There are many ways in which one could set up statistical methods to estimate the performance-function. I coin the terms performance-function-estimator and performance-function-estimation in this context. In the following I propose a particular performance-function-estimator: *performance-function-estimator- α* .

The overall strategy in performance-function-estimator- α is to first estimate the flatline-moment and the flatline-level. The term *flatline-estimation* is coined for this stage. Performance-function-estimator- α applies so-called sequential analysis for this purpose, a notion to be introduced later. After this stage has been completed, curve fitting is applied to determine the part of the performance-function that occurs before the flatline-moment. I call this the *growth-estimation*-stage, because it indicates the stage within which the performance of the candidate is (expected to be) growing. Preferentially (non-linear) regression analysis is used. The estimations of the flatline-estimation are used as constraints that must be satisfied by the growth-estimation.

*performance-
function-
estimator- α
flatline-
estimation
growth-
estimation*

II.5.6.8 Flatline-estimation

The algorithm designed for the flatline-estimation is sketch-wise described as follows. After each session played, the algorithm determines the *lowest* session number i for which holds that the performance-level can be estimated with a predefined level of *precision* if it only includes the performance-measurements from i to the latest played session (the 'tail' from i). If there is no such i , so if the required precision cannot be reached with the available sessions, the composition-record-player should play an additional session.

The motivation behind this algorithm is as follows. As stated before, the performance-function-measurement is modelled as a stochastic process that consists of the performance-function with noise added by uncertainty in measurement. This implies that the mean of this stochastic process is equal to the underlying performance-function. Now, if the performance-levels in the tail from a given i are so close together that they are indistinguishable from a sample from a fixed random variable, then it is quite reasonable to assume that this stochastic process has become completely or nearly

¹⁵The mentioned prevalent definition, which I encountered first, was one of the main reasons that this highly relevant subdiscipline eluded my attention for a long time. The intervals between measurements of the performance-function in an Orcoba-extension can be quite short. This is, for example, the case in the SWiFT-game (chapter III.8 (p. 245)), where they are less than a minute, to at most a few minutes apart. My first inclination was to investigate time series analysis, however, the majority of the literature in this field is about forecasting, and not about modelling past data, which is the central concern of longitudinal data analysis.

stationary (the probability distribution, including its mean, do not change anymore). This implies that the underlying performance-function has become constant: the flatline-level has been reached. Obviously, for this purpose, one has to choose for a sufficiently high predefined level of precision.

The following sections express this more precisely. It uses the terminology introduced in section II.5.6.5.3 (p. 93). The random experiment, in this context, is defined as the result of a performance-measurement at particular point in time. The outcome of the session is all data collected about this session. The stochastic variable X_{fst} maps this outcome to the score as it defined in the given context. ('fst' stands for fluency score translation session). Now, given the context of a player in a game, playing several game-sessions ($\text{game_ses}_0, \dots, \text{game_ses}_{n-1}$) in sequence. Stochastic process $X_{\text{fst}}(*)$ maps an index $i \in \mathbb{N}$ ($i < n$) to the stochastic variable that expresses the score achieved in the i th game-session, written as $X_{\text{fst}}(i)$. In other words, $X_{\text{fst}}(*)$ is the collection of random variables $\{X_{\text{fst}}(i) | 0 \leq i < n\}$ associated with a set of performance-sample-data.

Lets first assume that $X_{\text{fst}}(*)$'s distribution does not change over time. This is the default situation in common estimation theory. In this theory, a statistical parameter of a random variable is estimated based on a finite number of observations (a sample) (Lehmann and Casella, 2006).

The flatline-estimation can be tackled with common statistical estimation theory, when one rephrases it in terms of the notations just mentioned:

- The random variable X is equivalent to X_{fst} .
- The estimated population parameter θ is in this context the mean.
- A sample X_* from X (so, the random experiment of carrying out X n times.) is equivalent to stochastic process $X_{\text{fst}}(*)$. (This equivalence holds because of the assumption that $X_{\text{fst}}(*)$'s distribution does not change over time.)
- An observation x_* is equivalent to the time series $\{x_{\text{fst}}(*)\}$.

This paves the way to apply sequential analysis, a form of common estimation theory, to the flatline-estimation.

II.5.6.9 Applying sequential analysis

The assumption that the distribution of $X_{\text{fst}}(*)$ does not change, however, will in most cases not hold. Only after the flatline-moment is passed, the condition holds. First, two auxiliary functions are needed to define the procedure to estimate the flatline-moment. The first defines a function that takes a specified end-piece (tail) of

the game-sessions played so far and a desired confidence level, and maps these to a corresponding confidence interval:

Definition 16. $flat_lev_est_\alpha(x, i, \gamma) = conf_int((x_i, \dots, x_{n-1}), \gamma)$, where x is the game-session time series of a player collected so far, and $n = |x|$ (hence, $n - 1$ is the highest index of an observation within x) and γ the desired confidence level, and $conf_int$ a function that meets the condition given in definition 12 (p. 94), for example $conf_int_{balanced}$ from definition 14 (p. 96). ■

The second function uses $flat_lev_est_\alpha$ to determine the shortest tail of the game-sessions played so far that lie within a given acceptance set.

Definition 17.

$$\begin{aligned}
 &shortest_tail(x, \gamma) \\
 &= \max\{i \mid 0 \leq i < |x| \text{ and } ((x_i, \dots, x_{n-1}) \in accept_set(x_i, \dots, x_{n-1}))\},
 \end{aligned}
 \tag{II.5.20}$$

where x is the game-session time series of a player collected so far. ■

Moreover, an acceptance set needs to be chosen. For performance-function-estimator- α I have chosen for acceptance-by-percentage-deviation-from-mean ($accept_set_perc_dev$) as defined in definition 15 (p. 96).

The procedure is then described as:

Definition 18.

1. Let $i = 0$, and use it as an index for the observations made.
2. Start with doing the first observation $x_{fst}(0)$. Hence, in this stage the sample $\{x_{fst}(\ast)\} = (x_{fst}(0))$.
3. <VALIDATE>: Determine the shortest end-sequence that lies within the acceptance set, if any: $s = shortest_tail(\{x_{fst}(\ast)\}, \gamma)$.
4. If there is such a shortest end-sequence, so if s is defined, then terminate the procedure and return the estimations: $flatline_moment = i$ and $flatline_level = \{x_{fst}(\ast)\}$
5. Otherwise, increase i by one, and add a next observation $x_{fst}(i)$ to $\{x_{fst}(\ast)\}$. This leads to: $\{x_{fst}(\ast)\} = (x_{fst}(0), \dots, x_{fst}(i))$.
6. Repeat the procedure from step <VALIDATE>. ■

This completes the definition of the flatline-estimation.

II.5.6.10 Growth-estimation

As said before, the second stage of performance-function-estimator- α consists of the growth-estimation-stage (where the flatline-estimation is used as one of the constraints that must be satisfied by this growth-estimation). (Note that the second part of the performance-function has been determined in the first stage of the estimation procedure, and the first part is determined in the second stage.)

As stated before, regression analysis is preferred because it is based on a rigorous analysis of statistical errors (see section II.5.6.5.5 (p. 97)). This, however, requires a statistical model that is good enough. Such a model has not yet been created in this study at the time of writing. Therefore, at this stage I suffice with curve fitting that is less precise and rigorous but less demanding than regression analysis. This exploration can be used to create and further refine the statistical model. This is left as future work.

LOESS

locally weighted
polynomial re-
gression

Performance-function-estimator- α uses LOESS, also known as *locally weighted polynomial regression* (Cleveland, 1979; Cleveland and Devlin, 1988), as its method for curve fitting. My decision to choose it was informed by its widespread usage and inclusion in well-known statistical packages such as R. Future experimentation and fine-tuning, however, has to decide in which cases it is the best choice.

Moreover, there are other ways to presumably drastically improve the estimation process. They are put forward in section II.5.6.13 (p. 108).

II.5.6.11 Joining growth-estimation and flatline-estimation

The challenge that remains is how to join the two partial estimations, growth-estimation and flatline-estimation, into an overall estimation of the performance-function. In performance-function-estimator- α the following strategy is chosen. Divide the time-axis of the graph into three pieces: the first piece is equal to the growth-estimation, the middle piece joins the first and last piece smoothly (the join-graph), and the last piece is equal to the flatline-level. The middle piece runs from somewhat before the flatline-moment to the flatline-moment.

The concrete choice of the join-graph in performance-function-estimator- α is as follows.

Definition 19 (join-graph of performance-function-estimator- α). *Let grow be the function that resulted from the growth-estimation with performance-function-estimator- α (see section II.5.6.10). Let flat_mom and flat_lev respectively be the flatline-moment and the flatline-level estimated with performance-function-estimator- α . Now, define $t_{js} = c \times \text{flat_mom}$, where c is a fixed constant with a value from $[0, 1]$. For example, one could choose $c = 0.8$. Define point $p_{js} = (t_{js}, \text{grow}(t_{js}))$. Define the join-interval as $[t_{js}, \text{flat_mom}]$. Now, the join-graph join is defined on the join-interval and based*

on cubic spline interpolation (Schoenberg, 1946; Ahlberg et al., 1967; De Boor, 1978). Carry out cubic spline interpolation (Ahlberg et al., 1967) with 2 knots $k_0 = p_{j_s}$ and $k_1 = (flat_mom, flat_lev)$. The spline should satisfy the following continuity and ‘smoothness’ conditions. At $t = k_0$: $join(k_0) = grow(k_0)$, $join'(k_0) = grow'(k_0)$ and $join''(k_0) = grow''(k_0)$. At $t = k_1$: $join(k_1) = flat_lev$ and $join'(k_1) = join''(k_1) = 0$. This means that the spline consists of 1 interval (‘piece’) on which a cubic polynomial is defined, such that the additional conditions hold. ■

This leads to the concluding definition.

Definition 20 (performance·function·estimator· α). Let *grow*, *join*, p_{j_s} , *flat_mom* and *flat_lev* be as defined in definition 19 on the facing page. The performance·function·estimator· α for given performance·sample·data is defined as follows.

$$perf_estim_alpha(t) = \begin{cases} grow(t) & \text{if } t < t_{j_s}, \\ join(t) & \text{if } t_{j_s} \leq t < flat_mom \text{ (the join-interval)}, \\ flat_lev & \text{if } t \geq flat_mom. \end{cases}$$

Preliminary experimentation with joining graphs this way have been successfully carried out in R^{16} . (Not reported in this book.) Future work could include scripting this solution and integrating it into a software implementation of Or-evohut- α . ■

II.5.6.12 Future work join·graph

The join·graph in definition 20 is a first attempt to join the statistical models for the first and the last part of the performance·function·estimation. It is a predominantly heuristical approach based on smoothing. (It is also heuristical in the sense that the approach does not have a strongly principled basis.)

This heuristic probably requires further refinement. For example, the current growth·estimation procedure, could produce points that are higher than the flatline·level at $t < t_{j_s}$. This is in conflict with the assumption that the maximum is reached at the flatline·level (see section II.5.6.2.3 (p. 82)). If it is plausible that the assumption holds based on the concrete performance·sample·data, this problem could be fixed by slightly changing the join·graph heuristic. For example, it could be defined such that it disregards these points, if possible. It could do so by changing the size of the join·interval at its left side, instead of working with a fixed constant c (see definition 20).

¹⁶With thanks to Gerko Vink, expert in R.

Another question could be to establish a more rigorous join method that is based on a minimisation of statistical errors instead of, or in addition to smoothing. The question is, however, if such a method would perform better than the heuristical approach, or whether the increased precision is sufficient to be worth the additional effort.

II.5.6.13 Future work on performance function estimation

II.5.6.13.1 Improving growth-estimation with user interaction

The growth-estimation could benefit greatly from additional information provided by the player. For example, suppose that there is a sudden drop in performance from one session to another, then the system could ask the user if (s)he was distracted or that (s)he really experienced this challenge as more difficult. If the first is the case, then that may be a reason to discard this result.

Another important problem is that in the current approach to curve fitting, it is virtually impossible to get a reliable model. This is the approach in which a predefined class of functions is fitted to the performance data points, using only those points as source of information. A fictitious example may elucidate the problem:

1. In the first four sessions, $s_0 \dots s_4$, a player struggles to understand the challenge. His performance level remains fairly constant, at a low level.
2. During session s_5 , he experiences an 'aha-erlebnis'. This leads to a discontinuous jump in his performance level.
3. During $s_6 \dots s_{10}$ he does not experience an additional 'erlebnis', and his progress is gradual, following a sigmoid like curve, which is typical for this learning mode.
4. Consequently, in s_{11} , he has a new 'aha-erlebnis', but it requires some investment because he has to partially 'unlearn' what he has learnt so far.
5. Hence, in sessions $s_{11} \dots s_{13}$ he experiences a learning dip.
6. After that, in sessions $s_{14} \dots s_{20}$ he again gets into a 'gradual refinement' learning mode, following a sigmoid like curve, and reaches a flatline-level.

The problem could be solved with a piecewise function, so a function that consists of different subfunctions on different intervals. The statistical assumptions that determine the intervals and the functions is based on information that has to be obtained in addition to the performance data points. This information can be obtained by questioning the player after each session, with questions such as:

- “Were you distracted?”
- “Did you experience an ‘aha-erlebnis’ (a sudden understanding of something you did not understand before)?”
- etc.

Moreover, the changes in performance could be shown before the player is questioned. For example, a question be: “Your performance in the last session was quite somewhat lower than the sessions before. Was this caused by 1) a distraction 2) a question that you experienced to be more difficult than the previous ones.”

II.5.6.13.2 The Elo rating system

Future work may include exploring the integration of the Elo rating system and related systems into the performance-function-estimator. The Elo rating system (Elo, 1978), originally developed for chess, seems to have been embraced by quite some researchers in fields related to educational science, and in particular technologically enhanced learning and e-learning (Pelánek, 2016). At first sight, the Elo rating system seems not to be suitable as a performance-function-estimator because in its original form, it is a relative performance metric: in chess, it is determined by the outcome of matches between competing players. In other words, it is inferred from wins, losses, and draws against other players (Wikipedia contributors, 2018). In Or-evohut, and more generally in Challenge-Based-Orcobas, absolute performance measure are often required. However, the method can be generalised to an absolute measure with a trick (Pelánek, 2016). The trick consists of interpreting a test as a player, and interpreting the difficulty-level of the test as if it is the capability of that player. That way a player making a test, can be interpreted as a player playing against another player, which allows application of the Elo rating system. A problem that seems to remain is that the result of the tests in the scope of Or-evohut often can take more than the 3 values needed in chess (win, loss or draw). The Elo rating system, however, can also deal with a richer set of results. There are even approaches that integrate both the speed at which the learner produces the answer and the correctness of the answer into the Elo rating system (Klinkenberg et al., 2011). Such approaches could be applicable to formal-fluency, the central focus of part III (p. 131), in which these two factors are essential.

Moreover, Pelánek (2016) mentions that the Elo rating system is also well-suited for measuring the degree of a skill if it is changing during the tests, and that it is applicable to a wide range of capabilities without further adaptations, while its accuracy is quite good.

II.5.7 Or-evohut

Or-evohut- α

Or-evohut- α defined in this section, is an Or-evohut system that is intended as a baseline for comparison of Or-evohut systems. It is not intended to take into account all potential problems, but rather to help describing the advantages of extensions and alternative systems more easily. Moreover, whether a potential problem may manifest itself as a real problem in the implemented system, and also to what extent, can often only be determined by actually testing and applying such a baseline system. In most cases this cannot be determined by reasoning alone. *Or-evohut- α* 's definition uses the basic terminology and notations presented below.

Definition 21.

- *iteration*: an iteration of the evolutionary computation algorithm, so one pass of the repeated part of the algorithm (in other words: one generational cycle). As described in the blue print algorithm in section II.5.3.1 (p. 68), each iteration usually starts with selecting parents, and ends with selecting individuals for the next generation.
 - *i*: standard variable-name chosen to represent a generation, starting with $i = 0$.
 - Pop_i : the population (set of evo-individuals) after iteration i is completed
 - Par_i : the individuals in Pop_i that are selected to become the parents of individuals in Pop_{i+1} .
- potential-child*
potential-evo-individual
potential-child
- *potential-child = potential-evo-individual*: Given an evo-individual P that is selected to produce n children. Each of these children is a *potential-child*, and specifically, it is a *potential-child* of P . Synonymously, it is called a *potential-evo-individual*, a term which does not emphasise the relation it has with its parent. This notion is introduced to make it possible to talk about a child before it comes into existence. So, that is before the application of the variation operators (in this case: human editors) to its parent is completed. Note that this notion is not needed in ordinary evo-individual because there the offspring is created automatically, while in *Or-evohut- α* , they are created by humans in a process during which you want to be able to refer to the evo-individual in its becoming.
- potential-child-creator*
- *potential-child-creator*: A creator C of a *potential-child* is the player who has taken the role of creating it by editing its parent. A *potential-child* can be without creator, if no one has taken the role yet. In *Or-evohut- α* , a *potential-child* can have at most one creator.

- **child-creator:** *A creator of a child is the player who has created it by editing its parent.* child-creator
- **potential-child-realisation** *A potential-child C of parent P is realised, at the moment its child-creator completed working on it and publishes C . Of course, potential-child C loses its potential status at the moment of realisation. Note that while a potential-child goes through one or more changes, a realised child is fixed.* potential-child-realisation
- **branch:** *In its most general meaning, an Or-evohut branch is a line of descent of evo-individuals that are conceptually considered to form different versions of a single line of development. In Or-evohut- α this is typically the succession of evo-individuals created by the same human creator. Note, however, that the same creator may decide to start a new branch if a previous one is terminated.* branch
- **branch-continuation:** *an evo-individual is a branch-continuation if it is on the same branch as its parent. In other words, the evo-individual forms the continuation of the branch of its parent.* branch-continuation
- **branch-root:** *an evo-individual is a branch-root if it is on another branch than its parent. In other words, the evo-individual forms the root of a new branch.* branch-root
- **branch-creator:** *CR is the branch-creator of a branch B , if CR is the player who creates and/or created the children of the branch. Note that in Or-evohut- α , there is only one player per branch. So, if CR stops playing the game for good, B simply stops developing further.* branch-creator
- S_i the size of the population at iteration i .



II.5.7.1 Or-evohut branches

The most general meaning of an Or-evohut branch is that it is a line of descent of evo-individuals that are conceptually considered to form different versions within a single line of development. Compare this with what happens in the process of software development, for example the development of the open source word processor OpenOffice, in which subsequent version releases are considered to belong to the same overarching project. LibreOffice, however, has branched off of the OpenOffice branch as a separate, distinct branch, that is considered to be an application distinct from OpenOffice.

Note that the ‘branch’ notion, may be void of meaning from the perspective of the typical evolutionary computation systems¹⁷. The reason is that in the latter the variation operators are ‘random’, so there is no particular reason to assign a specific child a role different than the others, the role of being the ‘continuation’ of the branch of its parent, while the others are starting points of new branches.

This situation is different in Or-evohuts, because the variance operators are formed by human beings. A child-creator c may be regarding work on a potential-child as creating a new version within a single line of development that c has set into motion. In other words, he may be conceptually regarding the child, its parent and its future descendants as one single entity that he is developing (with the intent of improving it), just as the software developers’ perspective on different versions of LibreOffice.

This is closely related, if not identical to the paradox of the simultaneously singular and plural existence of a ‘changing entity’. For example, if a person is in the process of writing an article, this article is both a singular entity, and many different articles (each intermediate version being an article distinct from the others). The difference is that between (a kind of) *intension* and *extension* (Carnap, 1956): in the physical (extensional) realm, there are different articles, while intensionally they are all different versions of the same entity in flux.

The branch notion is not a purely theoretical concept. To the contrary, foremost it has important practical uses. First, a branch-creator, may also have an overarching strategy concerning the future directions of his branch. (Among others, he may implement ideas in a crude form in the child he is currently creating, with the intent to further extend that idea in future children on his branch if the crude implementation turns out to have promising results.). This overarching strategy is clearly tied to the branch as a whole, and not to a specific child. The concept is therefore needed to be able to talk about this dimension of the game. Second, the branch concept may work as a motivational instrument: creating a child is a one-time process, while developing a branch is potentially endless. This may create continued commitment among players, to make, or keep, their branch better than the other branches. The following sections describe Or-evohut- α ’s evolutionary human-computational algorithm step by step.

II.5.7.2 Iteration 0 (initialisation)

The size of the initial population (S_0) is 1, and there is one initial composition-record (composition-record α) which contains the bare minimum needed to tackle the challenge.

¹⁷Undoubtedly, there are evolutionary computation variants for which this does not hold.

II.5.7.3 Iteration 1 (expansion)

The second iteration, iteration 1, produces population Pop_1 as follows. Composition-record α is cloned once to be included in Pop_1 . The reason is that in iteration 0, it was the best evo-individual (it had no competition!). This will allow it to compete with the composition-records in Pop_1 . Then, a number of potential-children equal to S_1 are created. In iteration 1 these children are all children of composition-record α . There is no child which is the branch-continuation of composition-record α . Equivalently, all children are branch-root. Any player P who mastered the challenge, can sign up for becoming potential-child-creator (and becoming branch-creator of the associated branch). P becomes potential-child-creator if the number of assigned potential-children is still below S_1 .

For the remaining, the rules and activities described in section II.5.7.4.1 on the following page apply.

II.5.7.4 Iterations $i \geq 2$

The production of populations in the subsequent iterations is governed by the following rules. First, the population size is fixed for all generations, and equal to the size of Pop_1 (so, $S_i = S_1$ for all $i \geq 2$). The best λ (fixed in advance) evo-individuals from Pop_{i-1} are maintained in Pop_i (equivalently: selected to become part of Pop_i , not to be confused with parent selection, which means: the *potential-child* of the selected evo-individual is selected to become part of Pop_i). This is to guarantee that the best performing evo-individuals remain part of the population. Note that for these individuals, the fitness function does not have to be calculated again. The sole reason for transferring them to the new population, is that they can produce offspring in the iteration after that. In other words: those individuals do not only produce indirect offspring in later iterations (by grandparent relations), but also directly. Subsequently, $\mu (=S_i - \lambda)$ individuals are chosen using proportional selection and stochastic universal sampling with sigma-scaling of the objective fitness function (See section II.5.5 (p. 73)). The objective fitness function chosen for Or-evohut- α is SeqaNore. Child-creator application and assignment goes as follows. For each evo-individual i that is chosen at least once, the creator of i gets automatically assigned to create one of the children of i : the child that forms the continuation of the branch of i . The remaining available potential-children (equivalently: the parent selection) are announced to all candidate child-creators. Note that if i is selected more than once, these candidate child-creators can apply to become child-creator of one of i 's children. The principle is 'first come - first serve', so the first candidate child-creator has most of the options.

For the remaining, the rules and activities described in section II.5.7.4.1 on the

following page apply.

II.5.7.4.1 General iteration rules

General rules, that hold for all iterations $i \geq 1$ are the following:

- One branch per branch-creator: each branch-creator c is creator of exactly one branch at a time, no more and no less. Moreover, c can only become creator of a new branch, if c has ended working on a previous branch for good.
- One branch-creator per branch: there may only be one branch-creator active in the branch.
- Fixed iteration duration: the duration of each iteration is fixed to be I_{dur} . The intention is that all potential-children are completed before that duration expires. The motivational instruments described below are intended to promote the latter.
- Revocation of the child-creators status: if the duration of the iteration has expired and child-creator c has not realised all potential-children that were assigned to c , then, the child-creator-status of c will be revoked. If c already made changes to the potential-child pc , then pc is not deleted, but archived. Of course, the pc then loses its potential-child status. In the latter case, a new potential-child npc is created. Any candidate child-creator can apply to become child-creator of npc . Moreover, the I_{dur} is extended with some fixed time interval.
- Motivational instruments for releasing in time: these are instruments to motivate child-creators to realise their potential-child in time. In Or-evohut- α , these instruments are formed by:
 - (1) a human ‘generation iteration coordinator’.
 - (2) Appeal to social behaviour: players will receive reminder messages if they have not realised their potential-child, making an appeal to their social behaviour that they are delaying the game for others. Child-creators can also send each other motivational messages (particularly: a child-creator who realised his potential-child, may approach others to chase the pace).

II.5.8 Dealing with players who have trouble with a challenge

A problem that may occur is that a player has trouble mastering the challenge within a reasonable amount of time. If this is the case then at least the following negative effects may occur. (1) The player may simply give up and never return to the game. (2) The player may continue studying up to the point this has negative side effects (exhaustion, agenda conflicts etc.). This is a promising sign of motivation, however, it will in many cases lead to the player losing motivation after a few game sessions. Again, you may lose these promising players forever. (*End of list.*)

A solution for this problem may be provided by a panic button and a studying time-out of which the intended usage and behaviour is defined in the following subsections.

II.5.8.1 Design ingredient panic button

If a player p cannot solve the problem in an amount of time that is reasonable for p , the *panic button* can be pressed. If p does so, then the following will happen. First, the score will be the lowest possible (unsolved challenge). Second, the creators of the composition-record p used will be notified. They may help p in a chat session, to finally master the challenge. Third, if these creators are not available, every composition-record-user will be notified and may help p in a chat session. Fourth, p gets access to all composition-records. Giving access to all composition-records will increase the likelihood p succeeds, because p disposes over more studying material.

II.5.8.2 Design ingredient time-out studying time

Another provision to deal with a player p having difficulties with mastering the challenge is setting a maximum studying time for composition-records. If p exceeds this time, p will simply lose access to the composition-record. In that case the following will happen as well. If p still does not know how to solve the challenge, p is encouraged to ask for help by pressing the panic button. Note that it is not a good idea to automatically press the panic button for the player, because p might have studied sufficiently.

II.5.8.3 Negative side effects of the panic button

In the context of higher-order-Orcobas (see section II.4.6 (p. 61)), a problem may occur that relates to the help a player p receives after having pressed the panic button. p may receive help *beyond* the current subchallenge, giving p a head start in coping with

the dependent challenges in subsequent games. The point of higher-order-Orcoba is allowing modular composition-record assessment (see section II.4.6 (p. 61)). However, if in a chat session for the current subchallenge, aspects are explained which are the responsibility of a following, dependent, sub-Orcoba, the latter will be easier to cope with, regardless of the composition-record chosen for that subchallenge.

Solutions may be provided by including one or more of the following design ingredients. (1) Do a challenge test before the player starts studying the composition-record. Then you can do a comparative measure of progress. (2) Give a lower weight to the score of players who pressed the panic button before. (3) Record the complete chat session in which people helped the player. When a player has chosen a composition-record for a certain challenge, composition-record-players are prompted to scrutinise the previous chat sessions. If they find evidence that too much has been explained in chat sessions, the player's result may be disqualified to be incorporated in the overall score. (4) Simply doing nothing, and assuming that the advantage will be spread evenly over all composition-records, so that their relative scores are not in peril. (*End of list.*)

II.5.9 Conclusion and future work

II.5.9.1 Conclusion

This chapter has presented the most elaborate and central extensions of the Orcoba-Approach: the evolutionary inspired extensions. These extensions provide the following results with regard to the research questions posed in section II.4.1 (p. 49):

“

Research question 1.1.1 (motivating composition-record-developers).
How do you motivate, and continue to motivate composition-record-developers to create new composition-records or improve existing ones?

”

(Quoted from page 50.)

Conclusion 2 of research question 1.1.1. *In Or-evohut- α the generation cycles, each concluded with a performance evaluation, are fixed in length, and synchronised with the generation cycles of all other branches. (Note that this agrees with the standard practice in evolutionary computation.) This may have the following positive influences on motivation. First, the social pressure of other developers may prove to be a great motivational force. If a developer does not release his new child in time, the cycle has to be extended, also that of all other developers, who may urge the slow developer to chase the*

pace. Second, the competitive and comparative element will probably be a motivator: after each generation cycle each developer D will find out whether the newest version of D 's composition-record performs better or worse than that of D 's contenders, and also, than the previous versions of D 's composition-record. ■

“

Research question 1.1.2 (determining a quality metric for composition-records). *How do you effectively measure the quality of composition-records? This is a crucial instrument to determine whether there is indeed progress.*

(Quoted from page 50.)

”

Conclusion 2 of research question 1.1.2. *The measurement of the quality of composition-records, or, equivalently the ‘fitness evaluation’ from the perspective of Or-evohut, turns out to be far from trivial. It is a problem with multiple layers:*

1. *Stochastic fitness function: The (objective) fitness evaluation is of a stochastic nature: an estimation of the expected value of the score of teams playing with the composition-record is the outcome of the evaluation (section II.5.6.2.1 (p. 77)). The problem with this stochastic nature is that it contributes to the costliness of evaluations: the more precision one pursues, the larger the sample of scores one needs, thus, the more human resources one needs. In this section I have proposed sequential analysis, a statistical technique to be as economical as possible with the available human resources*
2. *Establishing the score of composition-record-users (e.g. a team using a specific composition-record) yields two subproblems:*
 - (1) *Dynamic performance: In some Evohuts, the fitness score of evo-individuals is defined in terms of the performance of persons who are interacting with that evo-individual. This performance is in most cases not static, but dynamic. Among other things, the performance will typically improve during the training for a certain period of time after training has commenced. It continues to do so provided there are no interruptions that are too long. The question that rises now is how to extract a fitness evaluation (a single real number) from the performance-function of several teams. I have proposed a way to do this: given a sample of performance-functions of different teams that is sufficiently large, first compress the performance-function of each individual team into one sensible real number (the performance-index), just*

as in the previous example, and then calculate the average performance-index (section II.5.6.2.2 (p. 78)). Determining a suitable definition of the performance-index, led to the important insight that there is no unambiguous choice that is best. It depends on what you want to achieve with the composition-record. Therefore, this section has made a start by defining a 'library' of indices in section II.5.6.3 (p. 82), while discussing what aspect of the performance they emphasise.

- (2) *Stochastic performance-function: the performance-function can never be known with complete certainty, because of uncertainties in measurements. Instead, one can try to estimate it based on the measurements. In fact, the measurement of the performance-function can be regarded as a stochastic process (section II.5.6.4 (p. 87)). This yields the following challenges. First, this is yet another aspect that contributes to the human resource intensiveness of the evaluation. Fortunately, by combining a number of assumptions based on cognitive psychology and educational science on the one hand, and so-called sequential analysis on the other hand, one can be at least be far more economical with human resources (section II.5.6.8 (p. 103)). Sequential analysis is used to estimate the flatline-level (the flatline-estimation). Then I proposed LOESS to estimate the growth-estimation-stage of the performance-function. This choice, however, is intended as a first version, and undoubtedly requires further fine-tuning. This is one of the reasons this chapter includes an overview to provide more insight into statistical techniques for curve fitting, amongst which regression analysis (section II.5.6.5.5 (p. 97) and section II.5.6.6 (p. 99)). These could serve as a point of departure for fine-tuning. Finally, I proposed cubic spline interpolation to connect the two models smoothly (section II.5.6.11 (p. 106)).*

■

“

Research question 1.1.3 (dealing with the exploration-exploitation trade-off). *How do you balance the trade-off between exploration and exploitation in the development of composition-records?*

”

(Quoted from page 50.)

Conclusion 1 of research question 1.1.3. *The answer consists of conclusion 1 of research question 1.1.3.1 on the facing page up to conclusion 1 of research question 1.1.3.4 (p. 121). These follow now.*

■

“

Research question 1.1.3.1 (dealing with finite developers effort). *The effort of the developers-collective is finite (per time unit), and therefore, has to be spent well. How do you effectively distribute the development effort over the composition-records?*

(Quoted from page 50.)

”

Conclusion 1 of research question 1.1.3.1. *The answer has two aspects: concentration and selection. Concentration in the sense that sufficient effort is spent per composition-record to make sufficient progress and selection in the sense of determining an effective selection of composition-records on which this effort is spent. One part of the solution consists of the fact that Or-evohut measures the quality of ‘active’ composition-records on sufficiently short, regular time-intervals in an objective, or at least ‘intersubjective’, way. Based on this measurement of quality (the fitness evaluation), Or-evohut adjusts its distribution of the collective development-effort. This proceeds as follows. In Or-evohut, there is a maximum number of composition-records that may be under active development (the population size) at any given time. In Or-evohut- α , there is even a fixed number (that acts both as a minimum and as a maximum). Assuming a fixed rate of available collective effort, choosing a suitable number will ensure sufficient concentration. Second, selection in Or-evohut- α is realised by an algorithm (the selection operator) that assigns a higher probability to successful composition-records to continue, and even fork into several branches (‘exploitation’). The probabilistic approach also implies that less successful composition-records have a chance to be continued, or even fork (‘exploration’).* ■

“

Research question 1.1.3.2 (dealing with finite assessment resources). *The amount of (human and other) resources to assess the quality of composition-records is finite per time unit. How do you distribute these resources effectively over the composition-records?*

(Quoted from page 51.)

”

Conclusion 1 of research question 1.1.3.2. *The answer to this is strongly related to the conclusion 1 of research question 1.1.3.1, and is also a matter of selection and concentration. With regard to concentration: the maximum population-size of composition-records in Or-evohuts, allows regulating the ‘consumption’ of fitness evaluations.*

The higher this maximum, the higher the consumption. By choosing a suitable maximum, and assuming that the rate of available fitness evaluations is approximately steady, one can ensure that each composition-record can be assessed within a given time-interval. ■

“

Research question 1.1.3.3 (preventing conservatism). *How do you prevent too much conservatism? Often, too much development effort is spent on maintaining the status quo, instead of exploring completely new, and possibly more fruitful directions.*

”

(Quoted from page 51.)

Conclusion 1 of research question 1.1.3.3. *As has been described in the conclusion 1 of research question 1.1.3.1 on the previous page, in Or-evohut- α , development-effort may only be spent on composition-records that are selected based on a neutral selection operator, which itself is based on an objective performance-measurement of that composition-record. This means that any new developer is treated equally in getting resources assigned to his or her composition-record. Note that the resources are fitness evaluations and developers working on children of that composition-record. In future variants of Or-evohut where more than one developer per composition-record is allowed, the system could distribute developers over composition-records in part based on the composition-record's performance. Contrast this with the situation in open source software development projects. Consider a popular open source project, with a large users and developers base. People who want to start a new branch of that project will have a hard time to acquire sufficient assessment resources (= human users and developers of the branch), regardless of the quality of that branch. (Note that in the terminology of the open source software world, the notion fork is often used in (much) the same meaning as branch within the context of this book.)* ■

“

Research question 1.1.3.4 (increasing equality in treatment of composition-records). *How do you give a fair chance to ideas within composition-records that may need time to mature before they lead to a better performance?*

”

(Quoted from page 51.)

Conclusion 1 of research question 1.1.3.4. *As stated in the conclusion 1 of research question 1.1.3.1 (p. 119), the selection operator works in a probabilistic way. This means that even composition-records that do not perform as well as others, still have a chance to produce offspring.* ■

II.5.9.2 Future work

I start with a note before listing potential future work. Full empirical experimentation with Orcobas is beyond the scope of the study reported in this book. It was beyond the available human resources to implement software for all of the complex systems proposed throughout this book. Choices had to be made. The resources have been mostly spent on developing the basis of the extensive code-base needed for the experiments with formal-fluency. Most of it has been dedicated to SWiFT-Focused (chapter III.8 (p. 245)), and the rest to SWiFT-Fixtal- α (chapter III.7 (p. 225)) and System- α -for-real-time-collective-formal-thinking (chapter III.5 (p. 197)). What is more, I have chosen to develop the software for SWiFT-Focused fully professionally. This way, it can serve as a base for further development in a larger open source community. Among other things, the software architecture, version management, dependency management, bug tracking, test procedures and deployment meet or even exceed professional standards. For example, the highly advanced Scala language and its rich typing system have been used and exploited as the base language, contributing to the architecture. This all took a considerable amount of time.

Fortunately, Or-evohut- α is integrated into the design of SWiFT. This means that a substantial part of the implementation of Or-evohut- α has already been realised. Among other things, participants can create and edit constitutions, fitness scores of constitutions are automatically determined based on the performance of players, performance-functions-data can be rendered graphically, there is advanced version management for constitutions and a simulation has been developed to test aspects of the way the system assigns fitness scores to constitutions (“JaraSimulation”). The advantage of the professional set-up of the current code-base is that it can be seamlessly extended in the future to commence full experimentation with Or-evohut- α , and other Orcoba-extensions. This completes my note about the empirical validation of Orcobas.

Future work could include the following topics:

1. Implement Or-evohut- α and experiment with it to validate its design. Generate ideas for future improved versions. An obvious choice is to first combine Or-evohut- α with SWiFT-Focused, because the latter is in an advanced stage of implementation. However, it should be applied to a wide range of capabilities to promote the development of the subsequent versions into sufficiently universal

tools for fostering hucolligence.

2. Do further R&D into performance-function-estimators. Among other things:
 - (a) Experiment with performance-function-estimator- α and if it turns out to be viable, create subsequent more refined versions of it.
 - (b) Refine the growth-estimation procedure with the suggestions in section II.5.6.13.1 (p. 108).
 - (c) Develop other performance-function-estimators and even other approaches to define the fitness evaluation, if these would improve the fitness evaluation in a certain respect for specific types of capabilities and conditions. For example, it may be interesting to experiment with the Elo rating system (Elo, 1978). Also see section II.5.6.13.2 (p. 109).
3. Develop a selection operator that, in addition to the performance-function, is based on explicit human rating of composition-records. Among other things, the opinion of experts may be of great value.
4. Instead of a fitness proportionate selection, one could also apply a rank based selection or a tournament based selection. It has been argued that the latter two have several advantages over the first (see respectively (Miller et al., 1995) and (Whitley, 1989)). Additionally, it may be so that it is easier to combine the latter two with explicit rating by people (Thierens, 2016).
5. In Or-evohut- α players cannot introduce new composition-records, but only incrementally change existing composition-records. Develop approaches that enable players to introduce completely new evo-individuals in later generation cycles. Among other things, this may accelerate the introduction of promising new ideas.
6. In Or-evohut- α composition-record-players can always see each other's composition-records. This can have both advantages and disadvantages. A disadvantage is that players may decide to copy parts of a competing composition-record. In doing so, in a sense, they circumvent the evolution process as it is guided explicitly by Or-evohut- α . After all, in a certain sense, players who are 'copying' contaminate their evolutionary branch with other branches. Among other things, this could have a negative effect on the diversity of the population and contribute to a premature convergence of the optimisation process to a possibly local optimum while there are optima that are better. (More 'exploitation' than 'exploration'.) An advantage of copying is that it could lead to new combinations of parts of existing composition-records that

may prove to be more effective than the existing combinations. In terms of evolutionary computation one would use the word *recombination*.

recombination

A potential solution that keeps into account both the mentioned advantage and the disadvantage is some form of regulated access. I coin the term *human-crossover* for this purpose. A composition-record-player who copies a part of a competing composition-record is very similar, if not identical to a form of crossover as it occurs in evolutionary computation. With regulated access one can try to achieve a balance that prevents premature convergence, while giving sufficient space to fruitful recombinations to occur.

human-crossover

7. In Or-evohut- α only one person is allowed to develop a branch. However, it may be beneficial, and often even preferential, to extend this to a team of a modest size. Among other things, the sense of collaboration may incentivise developers. Moreover, it is more in agreement with the Orcoba-Approach, in which the users of a composition-record are developers of it as well. An interesting idea is to allow, or even require composition-record-developers to occasionally transfer to another team in a regulated way. This can be regarded as a second form of human-crossover next to the aforementioned one. Here, not the ideas, but people are moving from one branch to another, bringing with them the ideas, but also the more intangible intentions and philosophies behind their former branch. This may enhance the aforementioned human-crossover-process. Team transfers, however, can have the same disadvantage as idea copying, and therefore these need to be regulated in some viable way.

Chapter II.6

Conclusion and future work

II.6.1 Conclusion

The research question posed in the introduction of this book part (section II.2.2 (p. 42)) and a summary of the corresponding research results achieved in this book part, including refinements of the research question, are as follows:

“

Research question 1 (the general quest). *How to foster human collective-intelligence (or briefly: hucolligence)?*

(Quoted from page 7.)

”

Conclusion 1 of research question 1. *This work is based on three principles for fostering hucolligence that have been expressed, and motivated, in the general introduction of this book: decentralised-development, institutionalised-self-reflection and human-artefact-co-evolution (section I.1.4.1.2 (p. 9)). The Orcoba-Approach has been developed in chapter II.3 (p. 43) as a top-level implementation of these three principles. The Orcoba-Approach is based on the concept of the Orcoba, which is a community-pursuing-a-capability, together with a record (composition-record) that allows the composition of their capability to be (at least partially) reproduced and examined. Moreover, the Orcoba's composition-record is open for the public for reuse, and is designed or supported by all members of the Orcoba. Additionally, the composition-record contains records pertaining both to the artefacts and the human capabilities. Chapter II.3 (p. 43) also motivated why this offers a solid top-level implementation of the given principles.*

The Orcoba-Approach is but a top-level solution. Actually realising the given principles and the overall goal to contribute to increased hucolligence, led me to formulate a further specialisation of this research question into research question 1.1 (p. 48), which is repeated below, followed by its result.

“

Research question 1.1 (Hucolligence with Orcobas). *What are extensions and variants of the Orcoba-Approach that significantly contribute to fostering hucolligence?*

”

(Quoted from page 48.)

Conclusion 1 of research question 1.1. *Major subchallenges that were established during the R&D process were: motivating composition-record-developers, determining a quality metric for composition-records, dealing with the exploration-exploitation trade-off and dealing with large composition-records. The extensions of the Orcoba-Approach introduced in this part deal effectively with these. These extensions are the Game-Based-Orcobas (motivating composition-record-developers), the Challenge-Based-Orcoba (determining a quality metric for composition-records), the higher-order-Orcobas (dealing with large composition-records) and, in particular, Or-evohut (dealing, in great detail, with all mentioned challenges, except for dealing with large composition-records). More detailed conclusions can be found in conclusion 1 of research question 1.1.1 (p. 63), conclusion 1 of research question 1.1.2 (p. 63) and the conclusion 1 of research question 1.1.4 (p. 64). Specifically, much attention has been paid to dealing with the exploration-exploitation trade-off, because of its great importance. This gave rise to a further refinement of this question into a number of sub-research questions.*

Or-evohut is an extension of both the Game-Based-Orcoba and the Challenge-Based-Orcoba. Its first base version, Or-evohut- α is the most detailed Orcoba-system presented in this work. Detailed conclusions regarding Or-evohut can be found in conclusion 2 of research question 1.1.1 (p. 116) to the conclusion 1 of research question 1.1.3 (p. 118).

II.6.2 Future work

II.6.2.1 Or-evohut

A summary of a selection of future work with regard to Or-evohut is as follows.

1. Further implement and validate the most developed evolutionary inspired extension of the Orcoba-Approach: Or-evohut- α , and develop further improvements and specialisations of Or-evohut. (Note that I would have preferred to include full experimentation with Or-evohut- α already in this stage, but due to limited human resources choices had to be made: see section II.5.9.2 (p. 121).)
2. Further develop performance-function-estimators to improve fitness functions for Or-evohut.
3. Experiment with human-crossover: the regulated exchange of ideas and people between different branches of evolution. Do this such that the danger of a premature convergence to sub-optimal composition-records is reduced, while the occurrence of new viable recombinations is stimulated.

For more details, see chapter II.5 (p. 65).

II.6.2.2 Higher-order-Orcobas

I believe that higher-order-Orcobas could make an essential contribution to the effectiveness of the Orcoba-Approach. Future work concerning higher-order-Orcobas may include the following:

1. Probably it will be a challenge to optimise the dependent-capability-graph. Among other things, it will not be clear in advance whether a given subcapability is chosen sufficiently small. It may be so that there are composition-records larger than the maximally allowed size that are significantly better than the best possible composition-record that meets the size requirement. Moreover, it may not be clear whether the dependencies between capabilities is indeed as strong as assumed. Therefore, create a human-technological system that facilitates the evolution of this break-up into subcapabilities. (Preferably, this system is an Orcoba as well – forming a kind of reflective Orcoba.) Automated statistical analysis could be used to investigate the actual strength of the dependencies. In this process, apply insights from the field of adaptive learning, such as performance factor analysis proposed by Pavlik Jr et al. (2009).
2. Short-term performance does not necessarily equate long-term learning effects (Bjork, 2017). Therefore, it may be useful to let composition-record-players revisit previous sub-Orcobas, and create sub-Orcobas that cover new combinations of existing capabilities. Develop a human-technological system to optimise these revisitations and combinations. Among other things, use insights from the theory of spacing effects (Ebbinghaus, 1885; Toppino and Gerbier, 2014).

Part III

Fostering formal·fluency

Chapter III.1

Overview and structure

This part treats one of the two central concerns of this book: fostering human-fluency-in-formal-languages-for-automated-deduction, briefly formal-fluency. First it places this concern in a broader context: both historically and ‘structurally’. Structurally, formal-fluency is part of a greater collective capability: an effective application of automated-deductive-reasoners to help people reason about their collective knowledge, in this work called integration-of-automated-deduction into society. Historically, an essential ingredient of the integration-of-automated-deduction into society can be traced all the way back to the ancient Greeks and Indians, who initiated the formalisation of deductive reasoning. In recent history this culminated into revolutionary breakthroughs such as the invention of predicate logic, which, combined with the invention of the computer, enabled an integration-of-automated-deduction into society. The historical context is treated in section III.2.2 (p. 136). This integration, however, is in several respects still in its infancy, because it is a multi-faceted and complex process. This part devotes a chapter to contribute to its further maturation, and in doing so, goes beyond the treatment of formal-fluency only. The core of the contribution is a proposal for instruments to measure (‘metricise’) the degree of the integration-of-automated-deduction into society (chapter III.3 (p. 151)). The rest of the part focuses on fostering formal-fluency. Because this work tries to apply the Orcoba-Approach developed in part II (p. 21), it first briefly investigates their mutual compatibility (chapter III.4 (p. 193)). After that it presents several specialised extensions of the Orcoba-Approach for the purpose of fostering formal-fluency: real-time-collective-formal-thinking and SWiFT (Semantic Web in Fast Translation). Then it presents further specialisations and variants of SWiFT: SWiFT-Full (chapter III.6 (p. 215)), SWiFT-Fixtal (chapter III.7 (p. 225)) and higher-order-SWiFT (which includes SWiFT-Focused) (chapter III.8 (p. 245)). The chapters also describe in detail

the relations between the several extensions, particularly in how they complement each other in the process of fostering formal-fluency.

Chapter III.2

Introduction

(Before starting this chapter, I would like to gently remind the reader of the call for feedback on page i.)

This part treats one of the two central concerns of this book: fostering human-fluency-in-formal-languages-for-automated-deduction, or more briefly formal-fluency. One reason formal-fluency is important is that it enables computers to help people reason deductively about their collective knowledge. Later, this work coins the phrase integration-of-automated-deduction (into human communities) for the latter. A more thought-provoking way to put it is that this type of fluency is part of an extension of the human mind and cognitive capabilities, to wit those capabilities that have to do with drawing conclusions that necessarily follow from already available knowledge. Just as hammers and screwdrivers are amplifiers for human muscular capabilities, this extension is an amplifier of certain human cognitive capabilities. Cybernetics pioneer Ashby (1956) speaks of *intelligence amplification* in this context.

III.2.1 Formal-fluency

I define formal-fluency as the ability to fluently express information in a deductive-formal-language. This definition leans on the use of the word ‘fluency’ as it is used in the context of natural languages. The dictionary defines ‘fluency’ as the “the ability to speak or write a language easily, well, and quickly” (Cambridge Dictionary, 2019). This is close to the common, layman, interpretation of fluency expressing overall language proficiency. A more operational, hence measurable, definition is in terms of accuracy and speed. Only when a person has attained a certain minimal level of accuracy and speed while expressing him or herself in a language, can he or she be

considered to be fluent.

III.2.1.1 ‘Fluency’ in natural languages

On closer inspection, however, the term ‘fluency’ (in natural language) seems to be a quite vague, ambiguous and multi-faceted notion. According to Chambers (1997), some communities, such as foreign language educators, do not include ‘accuracy’ in the definition of fluency. Fillmore (1979) discusses many different ways of being fluent in a language, and notes that “the word ‘fluency’ seems to cover a wide range of language abilities; these are best described with terms like articulateness, volubility, eloquence, wit, garrulousness, etc.” Riggerbach (1991) suggest, based on a study analysing the recordings of speakers who were labelled fluent or non-fluent by English instructors, “that fluency is a complex, high-order linguistic phenomenon and that intuitive judgments about fluency level . . . may take into account a wide range of linguistic phenomena.” Some other facets of fluency are that

- one can make a distinction between writing-, speaking- and reading-fluency;
- ‘fluency’ implies a great degree of effortlessness – the ability to express oneself without difficulty and exhaustion in a given language;
- the speed at which information is expressed, is minimally at a level that does not obstruct the flow of the thinking process too much.

Even if one limits fluency to accuracy and speed, one problem that remains is at what speed and what degree of accuracy one should consider someone ‘fluent’ in a language. A person who is considered to speak Luxembourgish ‘fluently’, will often not be able to translate his or her thoughts in real-time into this language, given the great speed at which thoughts can occur. Moreover, there are differences in speaking rates between fluent speakers of a language. Hence, it stands to reason that the minimally required speed and accuracy is determined by what is – on average, attainable by most native speakers when speaking with a low degree of effort.

III.2.1.2 ‘Fluency’ in formal languages

In the context of this book, I restrict formal-fluency to writing-fluency. After all, writing is the most obvious way to communicate with automated-deductive-reasoners.

The meaning of the word ‘fluency’ as it is used in the context of natural languages does not translate one-to-one to the way I use it in the context of formal-languages. Among other things, in formal-languages it is not only about accuracy in the sense that there are no mistakes in the expressions, but it is also about the degree to which

automated-deductive-reasoners can make use of the produced expressions. It is not trivial to define a metric for this purpose. In section III.3.5.2 (p. 174) I propose such a metric. Moreover, an important aspect of formal-fluency is modelling a given domain in a formal framework. In other words, in a sense one is continuously extending the formal-language to express information that pertains to certain domains (for more details, see section III.3.5.6 (p. 186)). In this work, I restrict formal-fluency to the dimensions speed, accuracy, the aforementioned degree to which automated-deductive-reasoner can be applied, and effortlessness.

Another question is at what degree of speed and accuracy one can talk about having attained ‘fluency’ in a formal-language. In this work, this question will be of secondary importance, because the focus is on progress in fluency. For this purpose, relative measures of fluency suffice. Typical questions I pose include: who has a greater degree of fluency than someone else, or does someone make progress? Hence, the emphasis is on fostering a greater degree of fluency, instead of attaining a predefined degree of fluency. After all, given the purpose of this book, the greater the rate at which information can be translated into a formal-language, the better – hence any gain in fluency is valuable.

III.2.1.2.1 Absolute degrees of formal-fluency

Nevertheless, a brief reflection on how to establish standards for minimal absolute degrees of proficiency in a formal-language that can be classified as ‘fluent’, can be useful. For one, one could try to, at the least, achieve a level that does not obstruct the flow of the thinking process too much. What that speed is, is to the best of my knowledge an open question. A safe bet, therefore, would be to achieve the same speed and accuracy as holds for fluent writing in natural languages. However, it is unlikely that many people would ever attain such a degree of fluency. Probably, a given piece of text will on average be translated more quickly into another natural language than in a formal-language by people proficient in the given languages – now and in the future, even after extensive training and refined training methods. Among other things, the aforementioned continuous process of extending the formal-language slows down the writing process in comparison with most of the natural language writing humans do.

However, it is a fact that also writing and speaking go much slower than the thinking process of probably most people. This can, among other things be implicitly derived from the great difference between reading speed (about 300 words per minute for a skilled reader) (Nation, 2009) and writing speed (about 13 words per minute for a skilled writer) (Bledsoe Jr, 2011). If we assume that the ratio between writing and thinking speed is so great as a consequence of physical and motoric limitations, it stands to reason that this ratio is partially circumstantial. Hence, this ratio can

probably stretched even further, while we would still perceive it as fluent.

As said, however, this book focuses on relative and not absolute degrees of formal-fluency. Therefore, with some minor exceptions, I will leave it at this reflection.

III.2.1.3 Why foster formal-fluency?

The amount of information represented in a formal-language currently may seem impressive. At the time of writing, presumably the biggest collection of public domain data-sets exists in the formal-language RDF+S, also called Linked Open Data. It currently consists of 1369 data-sets, amongst which the immense data-set DBpedia. <https://lod-cloud.net/#> maintains an overview of the current state of the collection (Abele et al., 2019).

However, this is only a minor part of the ever-growing body of information. To increase this fraction at a greater rate, it is crucial to expand the repertoire of available methods to translate information effectively into formal-languages.

The main methods in the current repertoire are as follows. First, we let experts, so-called knowledge engineers, do the work. The problem is that there are simply not enough knowledge engineers to keep up with the rate at which new information is produced. Note that this work also includes the never-ending necessity to extend the language used, e.g. in the form of introducing new ontologies, given the fact domains and paradigms are in a continuous state of evolution. Second, we use natural language processing techniques to let machines do the work for us. However, these techniques are in a primitive stage of development, and there is controversy around the question whether machines will ever be able to do the job properly. Third, we construct specialised (Web-) software that guides people with minimal or no training in creating such representations. I believe this only works for specialised use cases with specific types of information; otherwise training would be required. One of the several problems with specialised guidance is that for each specialised type of information, you need specialised software, and there are simply not enough programmers to cover the ever-growing range of types. The same problem holds for so-called Semantic Wikis (Bry et al., 2012): they typically provide support for formalising specific types of information.

Expanding the current repertoire with fostering formal-fluency in society holds the promise of accelerating the translation-rate to an unprecedented degree.

III.2.2 History and nature of automated-deduction

Formal-fluency derives its meaning from the context of automated-deduction. Therefore, knowledge of the history and nature of automated-deduction may contribute to

a deeper understanding of formal-fluency.

At the end of the 19th century sufficient scientific and philosophical work had been done to arrive at the amazing insight that *deduction*, which is a certain mode of reasoning, can be mathematically represented by mechanisms carrying out symbolic manipulations, and that these mechanisms *could* be implemented on a machine (*'automated-deduction'*). The mode of reasoning spoken of here is *deductive reasoning* (or *inference*), that is, drawing conclusions that necessarily follow from a set of given facts. A systematic treatment of deductive reasoning can be found as early as 300 BC in the works of the ancient Greek Aristotle, who is attributed with the invention of *syllogistic logic* (Smith, 2015). Independently, related research findings were made in the Indian Nyaya-school of thought, dating back as far back as 400 BC (Ganeri, 2004).

However, it took more than 2 millenia before the next major revolution in this field occurred. Around 1850, George Boole fundamentally extended the work of Aristotle and the Nyaya-school of thought, both of which he was acquainted with in his books "The Mathematical Analysis of Logic" (Boole, 1847) and "Laws of Thought" (Boole, 1854), creating a new logic system nowadays called Boole's algebra. This system was considerably more expressive than Aristotle's system, and Boole's system included a mathematically well-defined way to carry out inferences (by means of algebraic equation solving), setting a major step towards reliable automated-deduction. In a sense, Boole's work marks the start of the development of modern *symbolic logic*, the science concerned with investigating logical principles by means of formalised systems consisting of primitive symbols, combinations of these symbols, axioms, and rules of inference.

His work then matured through work done by, among others, Charles Saunders Peirce and Gottlob Frege, who independently made major contributions to the development of symbolic logic. For example, Peirce contributed to the further development of Boole's algebra into Boolean algebra, the version we use nowadays, and realised that Boolean algebra could be implemented by electrical switches, and Frege extended Boolean algebra to a much more powerful symbolic logic: *Begriffsschrift* (Frege, 1879). Predicate logic, a logic system based on, and equivalent to *Begriffsschrift*, is the present day, modern most expressive symbolic logic. Peirce, independently, developed an equivalent system a few years after Frege. Their work, unfortunately, remained largely unrecognised at the time of creation. Frege's predicate logic only became adopted as the fundamental symbolic logic, after dissemination by well-known scholars such as Whitehead, Wittgenstein, Carnap, Russell, Hilbert and Ackerman. The latter two published predicate logic in its modern form (Hilbert et al., 1950). Peirce's insight that Boolean algebra could be implemented on machines was later rediscovered (by among others Shannon and Sheffer) and led to the development of the first systems for automated-deduction, such as the logic theorist by among others Newell and Simon (1956) in the fifties.

*automated-
deduction
deductive reas-
oning
syllogistic logic*

automated-
deductive-
reasoner
formal-
language-for-
automated-
deduction
deductive-
formal-
language

This work uses the term *automated-deductive-reasoner* for these systems. Moreover, the term *formal-language-for-automated-deduction*, or more briefly *deductive-formal-language* is coined for the languages that each of these use. Hence, a *deductive-formal-language* is defined as a formal language for which one can construct algorithms (theorem provers), which when applied to expressions in the language perform semantic-preserving transformations on them, and for which this property can be proven by model-theoretic means (Dalen, 1997; Hodges, 2013). In classical logic the semantics that is preserved is typically the truth value. In this book, the term *formal-language* is used as an abbreviation for *deductive-formal-language*, unless locally specified otherwise. Note that the term *formal-language* is also used by others in related, yet different ways. This is, for example, the case in the field of formal language theory Chomsky (1959). The reader should keep this in mind.

partially-
automated-
deduction
proof assistant
fully-
automated-
deduction
automated the-
orem prover
automated-
reasoning

Not all deductions can be created effectively and timely with fully *automated-deduction*. In many cases, the computational complexity is too high. Often, one would have to wait for years, perhaps even thousands of years before a deduction is constructed fully automatically. This led to the development of systems in which humans and algorithms collaboratively construct deductions, such as Coq¹. Therefore, I will widen the notion ‘*automated-deduction*’ by adding two adjectives: *partially* or *fully* automated deduction. Systems for *partially-automated-deduction* (equivalently: *partially automated-deductive-reasoners*) include *proof assistants* (not to be confused with *interactive theorem provers*). Systems for *fully-automated-deduction* (equivalently: *fully automated-deductive-reasoners*) are also called *automated theorem provers* (Bibel, 2013). Automated-deduction itself is a subcategory of the more general *automated-reasoning*, which also includes automated forms of so-called inductive and abductive reasoning.

Research and development activities into symbolic logic and *automated-deduction* serve (and have served) several purposes, amongst which research into the foundations of mathematics (e.g. “What are the limits of mathematical proving?”, “How can mathematics be made more objective?”), philosophy (e.g. “What is the nature of logic?”), and interestingly, theology as well: Boole’s purposes included a better understanding of Biblical texts.

III.2.3 Amplifying human reasoning with automated-deduction

The following purpose is central to this book: using *automated-deduction* to help people reason about their (collective) knowledge. This is a major focus within, among

¹<https://coq.inria.fr/>

others, the research field of *knowledge representation and reasoning*. Why would such help for people mark a major leap in collective and individual knowledge discovery? There are two main reasons: speed and reliability.

knowledge representation and reasoning

Concerning speed: the amount of useful information representations that can be deduced manually (so, without the use of computers), per time-unit, from our collectively produced information is a small fraction of the total amount of such useful representations. This leads to long delays in arriving at conclusions, delaying the discovery of useful ones, or even arriving at conclusions too late, beyond the period they were useful. An example of a long delay is in the following fictitious example. Suppose there is little known about the genealogy of the medieval writer William Wright. Historian Hiromi discovers in an old letter in an archive in Cork that William had a (full) brother Bernard Wright, a well-known composer. Moreover, suppose a Norwegian anthropologist Anne, unaware of Hiromi and his discovery, discovers on a visit to an archive in Norway that Bernard Wright's father was Fabian Wright. From the combined information one *could* deduce that William's father was Fabian Wright. However, it may take years before the two pieces of information come to the attention of a human individual from the vast amount of historical information available world-wide. With automated-deduction this fact could have become apparent almost instantaneously.

An example of obsolescence is deducing information from sensory data from weather stations, or sensors related to human traffic. If it takes a person a year to deduce useful forecasting-conclusions from sensory input for the weather of tomorrow, this is clearly an obsolete conclusion.

It is interesting to note that deductions can become laborious for at least two reasons: the 'horizontal' size and the 'vertical' size of deductions. I define horizontal size as the number of premises (axioms) that have to be evaluated to make the deduction. Even if it is a simple deduction, if the number of premises is large, doing this manually will typically only disclose useful conclusions at a very low rate. This contributes to major delays in areas such as scientific discovery. (As in the aforementioned example around the writer William Wright.) The vertical size is the length of the deduction itself. A deduction may be complicated, even if it has relatively few premises. This is most notable in the area of mathematics-research: one of the characteristics of (current day mathematics) is that the number of axioms is limited, while the deductions can be extremely long, in particular when they are formalised. Many examples can be found in a list composed by Wiedijk (2015) with major mathematical problems for which a formal deduction was created with the assistance of automated-deduction. An example is the proof of the independence of Tarski's Euclidean axiom. A formal deduction created by Makarios (2012) occupies roughly an impressive 230 pages, while the premises fit on only one page of the same document. According to Barendregt (2014), it is to be expected that within a few

decades rather complicated proofs that will take humans many years to produce, will be discovered automatically in a fraction of that time.

Concerning reliability: the use of automated deduction can help discover errors made in existing deductions. Automated deduction produces formalised deductions that leave much less space for error than (purely) human deductions.

Facilitating large-scale and effective assistance by automated deductive reasoners of people reasoning over their knowledge is a multi-faceted and complex challenge, both along the technological and the human dimension. Among other things, there is need for:

- (1) (More) efficient reasoners.
- (2) Means to foster formal fluency.
- (3) Means to foster the human capability to directly, or indirectly, apply and use existing software tools for deduction,
- (4) standardisation of formal languages and/or effective automated translations between expression with an identical meaning. This includes a standardisation of their vocabularies, such as for example, when one would choose first-order logic as formal language, using the same constant to refer to the notion 'solar panel'.
- (5) Means to foster the human capability to effectively negotiate several trade-offs such as the one that exists between the expressivity of formal languages for deductions and the decidability/complexity of producing these deductions.
- (6) The meta-cognitive collective human capability to continuously improve and adjust the aforementioned aspects, such as the creation of new reasoners, the standardisation process of formal languages needed for deduction, etc. This is, among other things, needed because of various trade-offs. These trade-offs do not allow the creation of the 'best' solution that works in all cases (such as the 'best' reasoner, or the 'best' formal language). (See, for example, section III.3.5.4 (p. 180).)

*integration--
of automated--
deduction*

I coin the term *integration of automated deduction* (into human communities) for the aforementioned: it is the proficiency with which people in a given collective, on average, are able to find answers to their questions by direct or indirect application of automated deductive reasoners, provided that the information to answer these questions by deductive means exists in some form these techniques. The proficiency is *collective*, because the success of a person P finding answers depends on activities of others within the given collective. For example, if these others have expressed a

larger part of their knowledge correctly in a formal language, P 's success may be higher. Moreover, P 's success may be higher, if some of these others have written and shared good and efficient automated deductive reasoners. An example of an 'indirect' application of automated deductive reasoner is a person that benefits from automated deductive reasoners working 'under the hood' of an Internet service, while (s)he may lack any formal fluency.

A detailed and precise overview of the complete challenge to foster integration of automated deduction into human communities itself is far from trivial. An attempt to start the development of such an overview is to be found in chapter III.3 (p. 151). This book part then focuses on one aspect of this challenge, which is one of the central concerns of this work: means to foster the human capability to fluently represent information 'formally' (in this work called human fluency in deductive formal languages).

III.2.4 Terminology

Some terminology used throughout this part is as follows.

III.2.4.1 Existing formal languages

Formal languages referred to in this book are

- *Propositional calculus* (Dalen, 1997; Wikipedia, 2018b): Arguably the oldest formal language. The predecessors of the current, modern form, already date back to 300 BC to the work of Chrysippus (Bobzien, 2016). *propositional calculus*
- *First-order logic* (Dalen, 1997): The foundational formal language of modern science. It originates from the discipline of foundations of mathematics and has since then spread to many other disciplines related to research and development of formal languages. It is one of the most, if not the most dominant formal language, and has developed into the de facto standard. The latter manifests itself among other things in the fact that often properties of formal languages are shown by translating them into first-order logic, and that first-order logic is probably one of the most prevalent research instrument for research into formal languages. Also see section III.8.5.2 (p. 266). *first-order logic*
- *Higher-order logic* (Enderton, 2015): an extension of first-order logic where, in addition to quantifiers such as "for every object (in the universe of discourse)," one has quantifiers such as "for every property of objects (in the universe of discourse)." This augmentation of the language increases its expressive *higher-order logic*

strength, without adding new non-logical symbols, such as new predicate symbols.

- TPTP language*
 - *TPTP language* (Sutcliffe, 2010, 2017): A standard syntax for first-order logic and higher-order logic that is intended to be efficiently manipulable with computer programs.

- RDF*
RDFS
 - *RDF* and *RDFS* (Hayes, 2014): A pair of foundational formal-languages developed for the Semantic Web. “RDF extends the linking structure of the Web to use Uniform Resource Identifiers (URI) to name the relationship between things as well as the two ends of the link (this is usually referred to as a “triple”). [. . .] This linking structure forms a directed, labeled graph, where the edges represent the named link between two resources, represented by the graph nodes.” “RDFS provides a data-modelling vocabulary for RDF data. RDFS is an extension of the basic RDF vocabulary.” In this book the pair will be abbreviated with *RDF+S*.

- RDF+S*

- SPARQL*
 - *SPARQL* (W3C SPARQL Working Group, 2013): A query-language for *RDF+S*.

- description logic*
 - *Description logics* (DL) (Baader, 2003): “Description logics are a family of formal knowledge representation languages. Many DLs are more expressive than propositional calculus but less expressive than first-order logic”.

- OWL*
OWL2
Web Ontology Language
 - *OWL* (McGuinness, 2004) and *OWL2* (W3C OWL Working Group, 2012) (*Web Ontology Language*): Extensions of *RDF+S* based on description logics.

III.2.4.2 KR-base

- KR-base*

A *KR-base* is a database that contains knowledge representations expressed in a formal-language (its *KR-language*). The notion ‘database’ is used in a very broad sense: all collections of such representations stored digitally constitute a *KR-base*, even if they are not part of a ‘traditional’ database application. For example all digital documents in the formal-languages *RDF+S* and *TPTP language* that are accessible over the World Wide Web constitute a *KR-base*, in this case one that is distributed, heterogeneous and not centrally governed.

III.2.5 Research question refinement

This book part focuses on investigating the second of the overall research questions of this book, as defined in section I.1.3 (p. 7), and repeated below:

“

Research question 2 (the specific quest). *How to foster human-fluency-
in-formal-languages-for-automated-deduction (or briefly: formal-fluency)?*

”

(Quoted from page 7.)

Here this research question is further refined into the following questions.

First, formal-fluency is not a means to its own end. In this work, formal-fluency is considered a part of greater systems. In particular, this work has been motivated by the pursuit of systems that facilitate effective assistance by computers for people reasoning about parkedCommentByAlodytheir knowledge. Understanding this broader context may be important for fostering formal-fluency, and moreover it may be more satisfactory to have an overview of the overarching challenge of which fostering formal-fluency is a part. This gives rise to the following question:

Research question 2.1. *[Survey fostering integration-of-automated-deduction] What is a precise and operational definition of the systems that facilitate effective assistance of automated-deductive-reasoners for people reasoning about their knowledge, and a survey of subchallenges and (already) existing solutions to create and foster the development of these systems?*

[Conclusion(s): page 303] ■

Note that this research question is in fact not a sub-research-question, but a ‘super’-research question.

Two further questions are:

Research question 2.2. *[Fostering formal-fluency on a meta-level] What meta level specialised extensions of the Orcoba-Approach exist that can contribute to fostering (the emergence of good composition-records for) formal-fluency?*

[Conclusion(s): page 304] ■

Research question 2.3. *[Fostering formal-fluency on an object-level] What are (ingredients of) composition-records that contribute to increased formal-fluency?*

[Conclusion(s): page 307] ■

III.2.6 Feasibility

It may be useful to briefly reflect on the feasibility of fostering formal-fluency. Common objections to fostering formal-fluency and related topics include the following. (1) The approach is and will remain beyond the capability of (most) people. (2) The

approach will not be received with enthusiasm by most people. (3) The trade-off between an approach that saves cognitive workload and this approach will in most cases be decided in favour of the first.

My response to these objections, in corresponding order, are the following. (1) I think that it would be far too premature to make this statement. Humanity is currently at the dawn of the co-evolution of human capabilities to express information and information technology. Widespread usage of information technology in daily life is at most four decades old. Note that it took conventional literacy, another human capability of expressing information, many centuries to become prevalent. By 1750 almost 5000 years had elapsed since the first rudimentary appearance of the art of writing, yet more than 90 percent of the world's population had no access to this art (Cipolla, 1969). In the two following centuries we experienced an explosive growth of literacy. For example, in France literacy among females rose from about 30% in 1800 to above 95% in 1910 (Vincent, 2000). Moreover, this process was not trivial, as Vincent states: 'since at least the middle of the 18th century, states had learned from each other's progress in the struggle to disseminate the skills of reading and writing' (Vincent, 2000).

Based on this, I deem it reasonable to assume that the art of writing and the art of disseminating it are a result of a long evolution process, and I think that the same will hold for the form of expressing information that I am focusing on. It is important to state that I do not purport in any way that I am yet able to define a new form of literacy in its full maturity. Instead, with my work I hope to make a modest contribution to the acceleration of its evolution.

I derive a second argument from the study of artificial grammar learning, a field of psycho-linguistics. Interestingly, already in studies conducted in the 1950s and 1960s human test subjects were well capable of grammatically recognising and reproducing languages that were generated using artificial grammars (Chomsky, 1959) by means of implicit learning (Reber, 1967; Miller, 1958). Implicit learning of the syntax of the grammar of a language is learning its syntax without being consciously aware of the syntax rules of the language. Instead, one learns purely by exposure to example-expressions of the language. First-language acquisition by children takes place largely implicitly. I assume that implicit learnability of a language is highly correlated to the capability of reaching fluency in that language. Because deductive-formal-languages, such as first-order logic, are defined by means of such a formal grammar, it forms a strong argument in favour of the feasibility of formal-fluency. Contemporary research has further corroborated this by also incorporating neuroimaging techniques, that show that the brain-activity patterns that occur during natural language usage are very similar to formal-language usage (although there are also differences). For more about this, see section III.2.7.5 (p. 147).

Among other things based on these points, I deem it reasonable to assume that

the art of writing and the art of disseminating it are a result of a long evolution process, and I think that the same will hold for the form of expressing information that I am focusing on. It is important to state that I do not purport in any way that I am yet able to define a new form of literacy in its full maturity, instead, with my work I hope to make a modest contribution to the acceleration of its evolution. (2) & (3) When a person would reach a degree of fluency in the new art where the advantages outweigh the disadvantages, the motivation to apply it would most probably rise. In other words, there will simply be less of a trade-off. Again, compare this with conventional literacy capabilities. For most children who have just started elementary school, the cognitive burden of writing down things they want to remember during a conversation is so great, that it will seriously impair the conversation. In contrast, adults who have mastered the skill of writing after years of intensive training will easily complement their listening activity with making notes.

III.2.6.1 Small scale adoption as breeding place

A final important remark to put things into perspective is that I am not pursuing short-term mass adoption of a new way of sharing information, a focus that is prevalent among research for educational purposes or public social software such as Wikipedia. I think that a small scale adoption by specialised communities would already be very beneficial for those communities, and that within them a relatively mature state of this capability could be reached on a relatively short time scale. Moreover, these communities could serve as a breeding place for the maturation of the art and its didactics, potentially laying the foundations for a wider adoption in some distant future. I am explicitly focusing on the community of scientists, who are above averagely trained and motivated in sharing transparent and precise representations of their knowledge.

III.2.7 Related work

III.2.7.1 Systems to train formal fluency

Systems that explicitly target formal fluency are, to the best of my knowledge, quite rare. Andor is a game that appeared in 2018 to train fluency in classical logic.

III.2.7.2 The Semantic Web

The Semantic Web (Berners-Lee, 1999, 2001) is described by its founding organisation, the World Wide Web Consortium (W3C), as a Web of data – of dates and titles and part numbers and chemical properties and any other data one might conceive of.

Various technologies allow you to embed data in documents (such as RDF+S. Part of the Semantic Web is inference – reasoning over data through rules (W3C, 2018). Rephrased in the more general terminology of this book, the ‘Web of data’ is nothing but a representation of information in a specific set of deductive-formal-languages, and ‘the inferencing through rules’ is nothing but automated-deduction.

III.2.7.3 Controlled natural languages

controlled natural language A *controlled natural language* is a subset of natural language that can be accurately and efficiently processed by a computer, but is expressive enough to allow natural usage by non-specialists (Fuchs and Schwitter, 1995). An important category of controlled natural language is formed by those that provide an alternative syntax for a given formal-language. This way people ideally do not have to learn the formal-language. Although the syntax of a controlled natural language is a fragment of natural language, the semantics is defined formally. Of course, the syntax of the latter is chosen such that the formal semantics is as close as possible to the ‘natural’ semantics. An example is ACE used as an interface to description logic (Kaljurand and Fuchs, 2007).

Hence, this application of controlled natural language is a form of fostering formal-fluency. However, it approaches it from a different angle than the study in this part. Figuratively speaking: instead of training people to be able to jump high enough to cross the hurdle (express oneself in a given formal-language), it tries to lower these hurdles (change the syntax of the formal-language to make it easier for people to learn and use it).

I believe these approaches complement each other. I believe that controlled natural languages cannot replace formal syntax in the quest of integration-of-automated-deduction into society. There are intrinsic differences between truly natural language and formal-languages. By training people in formal-languages using a formal syntax, one can acquaint them with the peculiarities of the formal-language. Hiding these behind a natural language interface, although convenient, may also backfire. Among other things, a natural language interface does not prompt the user to switch to another mode of interpretation and another mode of thinking. This may lead to mistakes and a weaker understanding of the formalism. The syntax of a given formal-language can also be regarded as a carefully crafted instrument to express oneself very effectively in that formalism (Dijkstra, 1979). Such an instrument stimulates the user to exploit the power of the given formalism. Dijkstra suggests, for example, that the ancient Greek were not able to realise modern mathematical break-throughs, in part due to lacking sophisticated formal notations of current day mathematics. It is indeed noteworthy that programming languages, on average, have not become more resemblant to natural language over the past decades, but perhaps even less so.

For example, compare COBOL (Bemer, 1971) with the contemporary language Scala (Odersky et al., 2006.)

III.2.7.4 Semantic Wikis and related systems

Semantic Wikis are Wiki systems that allow its users to represent a part, or all of their information in a formal-language-for-automated-deduction (Bry et al., 2012). These representations are in principle created by laymen, and therefore, such wikis are related to fostering formal-fluency. Examples of such systems are WikiData (Vrandečić and Krötzsch, 2014), Semantic Media Wiki (Krötzsch et al., 2006), Loki (Nalepa, 2018) and OntoWiki (Frischmuth et al., 2015). Such wikis could benefit from the approach in this work to foster formal-fluency to improve people's ability to use these systems – a concern that, to the best of my knowledge – is not explicitly addressed by any of these systems.

Some Semantic Wikis prescribe guidelines and collaboration protocols on discussion pages. Rules or guidelines on content creation and collaboration are sometimes documented in separate pages. This is what they have in common with the approach in this book, the Orcoba-Approach when considering these guidelines and protocols as a capability-constitution. Nevertheless, they lack a sophisticated approach to guide the evolution of such guidelines, as present in the Orcoba-Approach.

III.2.7.5 Psycholinguistics and neuroscience

In the area of psycholinguistics and neuroscience quite some research has been done into questions that are related to fluency in languages. The focus is dominated by fluency in natural languages, but also includes formal-languages. An interesting middle position is formed by artificial grammar learning (Reber, 1967; Miller, 1958; Pothos, 2007), which is about the capability of people to grasp the syntax of languages with a formal grammar. These languages are typically devoid of meaning and of any other purpose than to investigate the capability of people to process certain syntactical structures. Although the name may suggest otherwise, they are commonly used to investigate natural language processing in humans. However, the results self-evidently also apply to formal-language processing in humans. Probably even more so, because formal-languages adhere to the rules of a formal grammar with certainty, while this is only a hypothesis for natural language.

Results from these research fields may be useful for fostering formal-fluency. Among other things, it may provide (1) more insight in the feasibility of pursuing formal-fluency, (2) clues about how to improve the learning strategy to acquire fluency and (3) ideas for favourable design decisions when creating new formal-languages. *Formal language theory* is a particular kind of study into the purely

*formal language
theory*

syntactical, or grammatical aspects of languages. Historically the emphasis was on studying natural language, although the name may suggest otherwise. It is particular in that it mathematically models a language as a set of strings over a finite alphabet, called a formal language. It is important to note that the notion ‘formal-language’ in this book is used in a more specialised sense. Among other things, in this book it typically includes a formal semantics. Therefore I use the terms syntactically formal language and formal language syntax theory in the sequel to prevent confusion. In formal language syntax theory, one abstracts from semantics, and focuses exclusively on syntax. In its modern form it was first described by Chomsky (1959). A central question posed by Chomsky was: given a set of increasingly strong models of computation, what are the languages that can be recognised by each of these computational models? He proposed the now famous set of models of computation by means of an increasingly expressive hierarchy of classes of formal grammars.

*Chomsky hier-
archy
formal gram-
mar*

These are nowadays known as the *Chomsky hierarchy*. A *formal grammar* is a set of mathematically defined (string) production rules that inductively define a (typically infinite) set of strings. Each class in the Chomsky hierarchy is equivalent to a model of computation. The most expressive one, the class of unrestricted grammars is equivalent to the strongest we know, Turing computability. Chomsky’s original question was about the structural (syntactical) properties of natural languages, in particular the computational-complexity (Hartmanis and Hopcroft, 1971) of these structures.

Collaborating with Chomsky, the psychologist Miller initiated the development of a methodology to do research into the unconscious (‘implicit learning’) capability of humans to learn and process syntactical structures: artificial grammar learning (Miller, 1958). The methodology consists of (1) Defining syntactically formal languages, in this context called ‘languages with an artificial grammar’. These languages, among other things, have varying degrees of computational complexity as defined in the Chomsky hierarchy. (2) Subjecting human test subjects to these language and seeing how well they learn to recognise or produce syntactically correct sentences without having been exposed to the rules. A good example is the study of Reber (1967). Interesting discoveries were made, amongst which was that human subjects turned out to be better able to master languages with a context-free grammar than languages with a regular grammar, while the first class is computationally harder. Fitch et al. (2012) coined this ‘Miller’s supra-regular hypothesis’.

In the past few decades study into artificial grammar learning has witnessed a revival by combining it with modern neuroscience, in particular neuroimaging techniques (Fitch et al., 2012).

Particularly noteworthy in this book’s context is the article of Friedrich and Friederici (2009) titled ‘Mathematical Logic in the Human Brain: Syntax’. This

article reports brain regions active when reading expressions in first-order logic, and implications for the efficiency of this process. The article departs from the premises derived from artificial grammar learning that both natural languages and formal-languages share the aspect of an underlying grammar which either generates hierarchically structured expressions or allows us to decide whether a sentence is syntactically correct or not. The authors state that the advantage of rule-based communication is commonly believed to be its efficiency and effectiveness. By subjecting 24 participants to an fMRI scan while reading first-order logic sentences, the authors provided brain-imaging evidence that the syntactic processing of first-order logic is indeed efficient and effective as a rule-based generation and decision process. These results support the idea that achieving fluency in formal-languages is feasible.

Siegmund et al. (2014) applied neuroimaging techniques to get insight into programming language understanding, among other things to demonstrate that these techniques can be used to gain insight into how improve the design of programming languages and tools for program understanding, and train programmers more efficiently. In their study, with 17 human test subjects, they found that for comprehending source code, five different brain regions become activated, which are associated with working memory, attention and, indeed, language processing. These may hint at what skills are favourable to train to improve programming skills. Among other things they conclude that the results indicate that, for learning programming, it may be beneficial to train working memory and language skills. Since programming languages can be considered to be a subclass of formal-languages that share quite some similarities with deductive-formal-languages, the results may also translate, to a certain extent, to the latter.

Chapter III.3

Integration of automated deduction into society

(Before starting this chapter, I would like to gently remind the reader of the call for feedback on page i.)

III.3.1 Introduction

Formal fluency is part of a greater collective capability: an effective application of automated deductive reasoners to help people reason about their collective knowledge. I coined the term *integration of automated deduction* (into a human community) for this greater capability in section III.2.3 (p. 138). Integration of automated deduction into society is a multi-faceted and complex challenge. Among many other aspects, it will benefit from improved formal fluency, the availability of efficient automated deductive reasoners, and widespread access to these automated deductive reasoners, including the knowledge and proficiency to apply them. One major purpose of this chapter is to contribute to the development of a better overview of the challenges and potential solutions associated with fostering integration of automated deduction into society. This chapter, therefore, has a broader scope than fostering formal fluency.

integration of automated deduction

The central instrument to create such an overview is the development of more precise metrics of the overall challenge of fostering integration of automated deduction.

This chapter presents the initial stage of this process. It both proposes an initial set of metrics and an initial and non-exhaustive overview of development challenges that are confirmed by these metrics. The word ‘initial’ should be emphasised here: to reach a mature stadium will probably take many years of fine-tuning as in a project with a narrower scope, but a very similar approach: the TPTP world (Sutcliffe, 2010, 2017).

It is important to note that integration of automated deduction into society itself is part of a yet broader collective capability, that of integration of automated reasoning into society. This capability includes other modes of automated reasoning, such as automated induction. Examples of existing formalisms and classes of formalisms are many-valued logic (Gottwald, 2017), fuzzy logic (Zadeh, 1988), probabilistic logic, Dempster-Shafer theory (Dempster, 1968; Shafer, 1976) and Bayesian networks (Darwiche, 2010). However, this chapter is limited to the given capability, amongst other things because of the scope of this book, and to take one step at a time before ‘scaling up’. I believe that the overall methodology presented here can be naturally applied to the broader collective capability.

III.3.1.1 Why a measurable definition matters

Why exactly does a much more measurable definition of integration of automated deduction matter? The following elucidates this by briefly reflecting on another challenge.

Every year a great contest is held in Australia: the World Solar Challenge¹. In it teams compete by creating a car solely powered by solar power. The team whose car crosses the set distance of about 3000 km most quickly wins. It is clear that the contest is held to stimulate the development of sustainable transportation. The beautiful thing of the formulation of the match is that it separates the way to reach the goal (the implementation) from the formulation of the goal, and formulates this goal in a very measurable way.² This measurable, yet ‘implementation-agnostic’ formulation of the goal gives the community of innovators a very valuable instrument. It unifies the best of two worlds: First, the innovators have far-reaching freedom to come up with ways to build a car. After all, the challenge does not prescribe the solution, but merely states the goal. Second, the different solutions can be compared very effectively and unambiguously, because the goal is defined in a measurable way. Among other things, this enables the innovators to (1) assess the degree to

¹<https://www.worldsolarchallenge.org>

²In fact, the actual challenge formulates a number of implementational aspects – for example that the vehicle should have four wheels. Nevertheless, these aspects could in principle be left out while still serving the point that is being made here. (Download link to the rules: https://www.worldsolarchallenge.org/files/1776_2017_bwsc_regulations_final_v2.pdf)

which their solution contains all ingredients necessary to score well given the goal. (2) effectively distribute their effort over the development of different ingredients needed to create a fully functioning car. It would, for example, be quite non-sensical for a team to dedicate all resources to developing a chassis for a car. Without the other parts of the car they will not win the race.

The following returns to the core concern of this chapter: integration of automated deduction into human communities. The latter challenge is, however, not formulated in a way that is measurable as the World Solar Challenge. Would it not be valuable if there would be such a formulation? This chapter provides an initiative to develop such a formulation.

III.3.1.2 Current status

There is certainly attention for increasing our overall understanding of what the challenge of integration of automated deduction into society constitutes. An example is the 1st ARCADE workshop³ during the Conference on Automated Deduction (CADE) of the year 2016 about “Automated Reasoning: Challenges, Applications, Directions, Exemplary Achievements”. Another is the ‘*Semantic Web Stack*’ proposed by Berners-Lee (Wikipedia, 2018c), which presents the overall architecture of the Semantic Web. To the best of my knowledge, the most impressive, mature and systematic and metric-based approach to fostering important parts of integration of automated deduction is the *TPTP world* developed by Sutcliffe (2010, 2017). Sutcliffe presents the TPTP world as a well-known and established infrastructure that supports research, development, and deployment of automated theorem proving (ATP) systems for classical logics. It is a widely adopted system that provides, among other things, several well-defined metrics to compare and evaluate ATP systems and has many of the features pursued in this chapter. The proposal in this chapter, can in a certain sense be regarded as an extension to the TPTP world.

Semantic Web Stack

TPTP world

However, to the best of my knowledge there exists neither a measurable, metric-based, definition of integration of automated deduction into society, nor a systematic approach to develop such a definition. Rather, I encountered formulations which already prescribe parts of the implementation, such as the Semantic Web Stack, formulations that are vague, or formulations that cover a part, but not all of the challenge, such as the TPTP world. Also, it seems that many innovators in this and related areas simply work on an ingredient of integration of automated deduction without questioning the progress of the development of the other parts of the greater system it is, should, or could become part of. For example, Hähnle (2016) argues that the automated-reasoning research community dedicates insufficient research

³<http://www.cs.man.ac.uk/~regerg/arcade/>

resources to user evaluations. At the ARCADE workshop 2016, a discussion topic included: “Have we done enough for automated reasoning to be used as a tool, where it is needed, for real-life applications?” This would be comparable to people working on a part of a solar car (for example a solar-car engine), without asking whether the other parts that make up a completely and well-functioning solar car are being worked on sufficiently. There is nothing wrong with such a specialistic focus, as long as it is complemented by others working to a sufficient degree on other parts of the picture.

It turned out that developing a measurable definition of integration of automated deduction is not an easy feat. A few general reflections on fostering any desired state of affairs in the world, whether it be the availability of an efficient solar car, or an effective integration of automated deduction, may shed some more light on this.

III.3.2 Reflections on fostering desired states of affairs

The purpose of this section is to shed some light on (1) how to effectively develop metrics, (2) how to foster adoption and effective application of metrics, and (3) potential caveats in the process.

III.3.2.1 Determining the right scope

Any actor who wants to foster certain states of affairs in the world has to determine a scope: to which state of affairs is he or she going to dedicate his or her development resources? The question is how to determine this scope effectively. In the first place, the scope is based on the intentions of the actor: why does the actor want to achieve a given state of affairs? *Example: the World Solar Challenge sets a well-defined scope. However, is this scope completely adequate given the intentions of the organisers to support the emergence and adoption of a more sustainable type of car? Among other things, the given scope does not take into account whether the solar car is comfortable for the average driver.* Another matter is that the intention of an actor can itself mostly also be considered as a scope within a yet more general intention. Perhaps, if one generalises the intentions even further, another scope within those general intentions would have been a better choice. *Example: The emergence and adoption of a more sustainable type of car itself is part of the yet more general intention of fostering the emergence and adoption of a more sustainable type of transportation. Perhaps it is a better idea to set another scope within that intention, for example wind-powered boats instead of solar cars.*

I will now define some terminology and make statements to further reflect on the essential matter of scope determination.

Statement 1 (generalisation). Most desired states of affairs are part of some greater or more general desired state of affairs.

Definition 1. A desired state of affairs is a *self-evidently-valuable-state-of-affairs* if its value is self-evident and does not need further explanation. A desired state of affairs is a *minimally-self-evidently-valuable-state-of-affairs*, if any state of affairs that is less general than it, is not a *self-evidently-valuable-state-of-affairs*.

If an actor seeks to make a positive contribution to the world by pursuing a certain state of affairs, then I believe it to be a valuable practice to repeatedly answer the ‘why do you want this’ question, until reaching the state of affairs for which it is obvious that its pursuit is worth it without any further explanation – the *minimally-self-evidently-valuable-state-of-affairs*. Then all involved actors can use this state of affairs to mutually balance and coordinate all efforts. If statement 1 is true, such a climb up the ‘generalisation’ ladder is in most cases possible. *Example: Why does the World Solar Challenge committee want to foster the emergence of solar cars that can travel 3000 km most quickly? Because they want to stimulate the emergence of a sustainable car. Why do they want the latter? Because they want society to make a transition to a more sustainable means of transportation. Why do they want the latter? Because they want society to become more sustainable. The latter will, from the viewpoint of probably most people from the Western culture be interpreted as something that does not need further explanation.* The scope ‘*integration-of-automated-deduction into society*’ itself also has been a result of making this climb partially. After all, climbing up the ladder from the scope of fostering formal-fluency, or the construction of automated-deductive-reasoners for knowledge representation and reasoning systems, may all lead to this more general scope. However, the ladder can be climbed up further – why do we want an *integration-of-automated-deduction into society*? A yet more general scope is the *integration-of-automated-reasoning into society*. However, why do we want that? Because of the fact we want to help people to answer questions based on the knowledge they have already acquired. From the viewpoint of most in the Western culture, probably we have reached the point that does not need further explanation.

However, I believe that the pursued states of affairs that do not need any further explanation depend on the culture, as expressed in statement 2.

Statement 2 (self-evidence is relative). Whether the value of a state of affairs is self-evident, is mostly relative. It is, for example, relative to a culture or subculture.

In reality almost no desired state of affairs is a *self-evidently-valuable-state-of-affairs* from any point of view. For subcultures, such as for certain subcommunities of

computer-scientists the widest ‘self-evident’ scope an actor reaches for why to foster integration-of-automated-deduction may be ‘to allow computers to help people to find answers to their questions’. From this scope, he or she may overlook completely different options to reach the same goal from a yet broader scope. *Example: What if we discover that telepathy turns out to exist and can be cultivated to such a degree that in combination with human reasoning it would be able to beat any computer-based approach?*

The ‘ideal’ self-evidently-valuable-state-of-affairs is often not reached by many of the involved actors, for better or worse reasons. The following reflects on several major reasons. The first reasons discussed here are related to problems surrounding the definition of adequate metrics.

III.3.2.2 Trade-offs in defining and applying metrics

III.3.2.2.1 Types of metrics

metric

In this context, a *metric* is a method that assigns a quantitative measure to a state of affairs that expresses to what degree it meets certain conditions. This work distinguishes three types of metrics: complete-metric, operational-metric and well-defined-metric. Before explaining these metrics, the following first introduces another notion required to present these explanations. A metric mostly does not stand on its own. In this context, the metric is regarded as an instrument to measure the degree to which a given state of affairs satisfies the intention of an actor. This definition may seem somewhat circular, because one could argue that the intention of the actor is what is measured by means of the metric. This circularity can be overcome by regarding the actual intention of the actor as something that can exist and evolve beyond the realm of language, something that exists in the mind of the actor. The metric can then be considered as a way to make this intention to a greater or lesser degree precise in language. This is closely related to what mathematicians try to achieve when formalising intuitive concepts, such as the notion ‘algorithm’ has been formalised with the definition of the ‘Turing machine’ (Turing, 1936). A difference, however, is that I allow less precision than in mathematical formalisations, for reasons that will be explained in this section. To establish whether a metric coincides with the actor’s intention, often an ongoing reflection is needed, amongst others by investigating whether specific states of affairs with a high score according the metric indeed are recognised by the actor as meeting his or her intention. Complete certainty in this is, often, unattainable. This is analogous to the situation in mathematics, where, for example, the statement that the formalisation of the notion ‘algorithm’ with ‘Turing machines’ is indeed adequate, the Church-Turing thesis (Copeland, 2017), cannot be proven mathematically. This means there is a possibility, however

improbable, that some day someone finds something that should be classified as an algorithm, that cannot be implemented as a Turing machine.

This information allows the definition of different properties of metrics. A *complete-metric* is a metric that ‘captures’ all aspects of the intention of the actor in bringing about a certain state of affairs. In other words, if a given state of affairs receives a perfect score, the state of affairs completely satisfies the actor’s intentions. *Examples: (1) For the intention to foster the emergence of a woman that can run the 100 meter as fast as a woman possibly can, the ‘women’s 100 meter’ is a complete-metric. This metric is dependent on the intention of the actor. (2) If one’s intention is to foster the emergence of women with a superb overall athletic capability, the women’s 100 meter cannot be considered to be complete. (3) Similarly, in the scope of a well-functioning solar car for everyday usage, the metric used in the World Solar Challenge cannot be considered to be complete. Among other things, the comfort that will be demanded by an average driver is not captured by the given metric.*

A *well-defined-metric* is a metric that is well-defined in the sense that the measure of a given state of affairs assigned by the metric is defined unambiguously. One of the greatest degrees of well-definedness can be reached by means of mathematical definitions. For example, computational-complexity as defined by Hartmanis and Hopcroft (1971) is a well-defined-metric. The opposite of a well-defined-metric is a *vague-metric*.

An *operational-metric* is a metric that can be applied in practice. For example, the World Solar Challenge and the women’s 100 meter each define such a metric. Both metrics can be used to determine the degree to which a given state of affairs complies with it. An operational-metric is typically also highly well-defined, but the converse does not have to hold. For example, the metric “the number of symbols that have been written by humans up to this date” is quite well-defined but far from operational.

Ideally, a metric is complete, well-defined and operational. However, I believe that such a metric often does not exist because of trade-offs between these properties. *Example: a metric to capture the intention behind the description “a solar car that can be used as a means for personal transportation,” is not completely covered by the metric formulated in the World Solar Challenge, while the description just given can be considered as a complete, yet very vague-metric.*

III.3.2.2.2 Caveat: settling for a well-defined-metric

Probably more often than not, the minimally-self-evidently-valuable-state-of-affairs does not allow the definition of a well-defined and complete metric, let alone an operational-metric. One caveat associated with this is that actors may settle for a scope that allows a well-defined or operational metric, but that is not complete. This

means that they are pursuing a state of affairs that does not completely align with their deeper intentions, and they may lose sight of the latter. This probably often happens unconsciously. This is a well-known and actual problem, for example in education, where operational metrics for performance of educational institutions are, according to some, used over-abundantly (Colyvas, 2012).

I think that the most fruitful approach is to use metrics as instruments that require ongoing reflection and evolution guided by the community using them. Among other things, one can overcome the vagueness of a complete metric by specialising parts of it with well-defined or even operational metrics. However, this process should be ongoing, by continuous reflection on the question whether the specialisations are sufficiently adequate, and changing them as understanding evolves.

III.3.2.3 Control

Another essential topic that may provide insights to effectively develop a measurable definition is that of *control*: to what extent do actors have control over the factors that help them reach a given state of affairs?

A common caveat is that actors may tend to settle for a scope over which they can exert great control, and in doing so, never reach the view-point of a self-evidently-valuable-state-of-affairs. I think it is reasonable to assume that this process often takes place unconsciously. A few reflections may help gaining somewhat more insight in this matter.

III.3.2.4 Resource limitations

A pursued state of affairs will in most cases contain aspects that are developed by *other* actors. *Example: A solar car developer, may, for example limit his choice for wheels to those available on the market – probably even if he envisions a wheel that he thinks to be better than any that is available. However, the wheel manufacturers who created them, clearly regards the wheels as systems the design of which is under their control to a great extent.* The reasons for ‘importing’ aspects designed by others include, among other things, the limitations in resources, such as time, and of capabilities of any actor. An individual solar car builder would need many life times to create a solar car, would he need to design and build all aspects of a solar car, the engine, the solar panels, the batteries, the wheels, the glass, the paint and so forth from scratch.

Another factor is the social position of the actor. The actor may be obstructed or assisted by others to realise certain parts of a state of affairs.

III.3.2.5 Decentralised control

States of affairs can be developed in a decentralised or a centralised way, or any degree in between these two extremes. In a centralised development process there is a single actor who controls and brings about all aspects that can be controlled by human beings. In a decentralised design process there are several actors and other extraneous elements that bring about different aspects relatively independently from each other, and often not with identical goals. In reality, it may be obvious that no design process is fully centralised. Decentralisation of control can have different causes. One of these is that a state of affairs may itself contain aspects that are controlled or influenced by other actors. *Example: if the intention of an actor is to increase sustainable transportation in the Netherlands, the actor has to keep into account that there are many other actors that have a say about this, such as potential buyers of sustainable transportation and political parties.* Another reason for decentralisation is the aforementioned limitation in resources of individual actors, necessitating collaboration (see section III.3.2.4 on the preceding page).

Often it is the case that the minimally self-evidently valuable state of affairs will be of such a size and complexity that the process of its development is highly decentralised. A caveat here is that actors may settle for a much narrower state of affairs to be able to benefit from a greater level of centralised control from their part. Again this may often be a process that takes place unconsciously.

An important implication for the development of desired states of affairs is that it could benefit from persuading other actors to work towards a common goal. A crucial instrument in this process is the definition of a clear definition of this common goal, which is exactly what is achieved with metrics.

In general, the development of integration of automated deduction into society as a whole is highly decentralised.

III.3.2.6 Self-similar state of affairs

Some states of affairs exhibit a self-similar structure. A state of affairs may consist of sub-states of affairs that have, to an important degree, the same nature and structure as the state of affairs itself. An example is formed by ecosystems as we know them from biology. The Netherlands has an ecosystem, which is part of Europe's ecosystem, which, in its turn, is part of the Earth's ecosystem. Moreover, the sub-states of affairs may also interact laterally with each other in a networked structure. This is the case for ecosystems, too. The ecosystem of the Netherlands does not only form a part of the ecosystem of Europe, but, among other things, interacts with other ecosystems in Europe, such as that of Germany and Belgium. Another good example is simply acquired by changing the word 'ecosystem' into 'economy' in the aforementioned

examples.

Integration of automated deduction belongs to this class of related states of affairs, so states of affairs that are self-similar and networked. *Example: one could consider the integration of automated deduction into the University of Otago, as part of the integration of automated deduction in New Zealand, and the latter as part of the integration of automated deduction into the world as a whole.* Moreover, the integrations of automated deduction may also interact laterally with each other in a networked structure.

The observation of the self-similar structure of integration of automated deduction into society has an important strategic value. The scope of this chapter has been formulated on an ambitious level: integration of automated deduction into society, which could be interpreted as the entire human world population. Attaining a high level of such an integration, however, is obviously an enterprise of a great complexity and size. To bring the challenge back to a manageable size, the scope can be narrowed down to smaller, yet considerably self-similar, scales: for example the scale of fostering integration of automated deduction into human communities of (much) smaller sizes, such as research teams, schools, universities and departments of organisations. The advantages include: (1) these smaller communities are direct constituents of larger communities, and therefore directly contribute to fostering it on the larger scale as well, (2) lessons learnt from a limited number of smaller communities, can be used to fostering it in other communities much more efficiently. Compare this with the typical government strategy to adopting new policies, these are often first tried and improved in a few municipalities before they are scaled up. (3) The smaller communities often allow a more centralised control (see section III.3.2.5 on the preceding page), which adds to their manageability.

III.3.2.7 Commitment

Another mechanism is that of commitment. For better or for worse reasons, an actor might be committed to the realisation of a certain state of affairs. A few notions may help further reflection on this topic.

Definition 22. *First, every state of affairs is part of a class of states of affairs. Each state of affairs belonging to this class will, from the perspective of the actors involved, be considered to be equivalent or similar.* Example: The notion ‘solar car’ is an example of such a class. It does not describe one specific state of affairs, but a range. After all, a solar car with any reasonable number square feet of solar panels will be regarded as a solar car, while they do not constitute identical states of affairs. *Note that this is the same notion generalised to states of affairs that was also mentioned in the scope of ‘classes of systems’ in section II.4.2 (p. 52)* ■

The degree to which a state of affairs is intrinsically valued depends on the actor. *Example: the emergence of a solar car may be valued by many, but not by all.* These degrees of *value* are typically further differentiated within any class of states of affairs. *Example: some may value the emergence of solar cars that are faster over those that are more environmentally friendly (if this would turn out to be a trade-off), and others may prefer the opposite.* Perceived value, self-evidently, is a major driver for intrinsic motivation of an actor to commit to fostering a certain state of affairs.

Another source of commitment is formed by the amount of resources that an actor already has invested in the partial realisation of a state of affairs. The higher this amount, the more an actor may experience a mental barrier to change course and pursue another, alternative state of affairs. This may even be the case if the actor finds out that the alternative may have a higher intrinsic value for the actor. *Example: if Akwasi spent a year on developing a solar car, he may not be so willing to change his design and start all over again, even if it would result in a better car and a lower total effort.* This is known as the *sunk-cost fallacy* (Arkes and Blumer, 1985).

Another form of commitment is social commitment: an actor may have made promises to others to commit resources to the realisation of a certain state of affairs. Such a motivation may be extrinsic, for example when the actor works as a contractor for a client. The effect of this can work both ways – if the client makes a well-informed choice concerning the state of affairs to pursue this works well. If the client fails to do so, then the process is even exacerbated if the contractor merely carries out the assignment without further questioning its broader purpose.

III.3.2.8 Trade-offs in pursuing states of affairs

Typically there will be trade-offs between states of affairs: the pursuit of a given state of affairs may go at the expense of another one. This holds for both trade-offs within a class of states of affairs, and between unrelated ones. There is a relation between trade-offs and the degree of value that a given actor attributes to them. If two states of affairs are both highly valued, but in a trade-off relation, the actor has to balance these trade-offs. *Example: Suppose, Amaruq dreams to own and operate a solar-powered car. However, living in Greenland, the climate is very unsuitable for that purpose. It would be best to emigrate to Australia. However, he wants stay close to his family as well. The latter is another state of affairs he desires. Hence, he decides to live in Sweden every summer, where the sun is somewhat stronger.*

The trade-off relations that typically occur imply that the pursuit of the most desirable state of affairs is an incessant process. One caveat here is that actors may tend to settle for a well-developed state of affairs, even if evolving circumstances suggest to balance trade-offs differently.

III.3.3 Approach

Because of the many trade-offs, the decentralised and multi-faceted nature of integration of automated deduction into society, such an integration can never be considered to reach a state of absolute completion. Rather, this integration should probably be considered as a continuous work in progress where, using adequate metrics, one can monitor the degree to which it progresses (or declines) and what the effects of different potential interventions will be, or past interventions have been. Compare this with the development of an economy in a country. Surely, there are economies which can be considered to be underdeveloped. However, any economy is a continuous work in progress and innovation, even if it has reached a high level of development.

This survey has been set up with that in mind. It goes beyond presenting the challenge as if it were a puzzle. For a puzzle it would be sufficient to present the already existing pieces and an overview of the pieces that yet have to be developed to complete it. An example of this approach is Berners-Lee's 'Semantic Web Stack' (Wikipedia, 2018c). In a sense, however, the challenge is much more like playing chess than completing a puzzle. Although certain positions have a well-known solution and can be laid out in advance, this is not generally the case. A book about chess, therefore, should also lay out meta-strategies to help players identify the type of problems they can encounter and how to cope with them. Many actual decisions, however, have to be made during the game depending on how it unfolds. Another comparison, more closely related to the topic of this chapter, is writing a book about the process of designing bicycles instead of the design of a specific bicycle. Certain aspects will be the same for almost all bicycles, but many other aspects may differ depending on the specific purpose of the bicycle, available resources and limitations.

The survey does so by including a reflection on the process of development, and not only a final 'product'. Major questions such as what are considerations and trade-offs that may or will occur during development and how could one balance these? The bicycle designing process will be used as a running example in the following. This requires a few things, described in section III.3.3.1.

In detail, presenting the challenges within fostering integration of automated deduction into society is approached by providing an insight in the main challenges, the subchallenges and potential solutions. Section III.3.3.1 explains the structure of the survey.

III.3.3.1 Structure of the survey

This section describes the structure of the survey and uses the bicycle designing process as a running example. In the explanation, the word 'system' in a broader sense is used instead of 'state of affairs'.

The treatment of each set of subchallenges contains the following elements: the definition of a focus, the definition of requirements, the definition of metrics, relevant-assumptions, relevant properties and development decisions, in addition to examples. Each of these are elements are elucidated in the following.

- *Focus*: if applicable, formulation of a focus within the development of the system in this iteration. If it is possible to tackle the whole design problem in one go, only one iteration is needed *focus*
- *Relevant-assumptions*: if applicable, definition of a number of assumptions regarding properties of the universe, used in this iteration. Some of these may be simplifying assumptions. These do not hold in reality, or only under specific conditions. They are intended to isolate the challenge addressed in the focus of this iteration better. This allows a better and sharper explanation of that challenge. In the survey a set of relevant-assumption, or *assumption-set*, may be designated a label to easily refer to it. Moreover, by replacing these assumption-sets at several iterations with more realistic assumptions, more complex challenges can be revealed in a more gradual built-up in level of difficulty. *relevant-assumption*
assumption-set
- *Properties*: isolation of relevant properties (of the ‘universe’ and of development decisions taken so far) that follow from the relevant-assumptions generally accepted assumptions. The properties are ‘relevant’ if they are essential for defining the metrics and (further) relevant development decisions. *Example: Objects moving through air, so also a bicycle and its cyclist, experience air resistance; Newton’s law determines the amount of force needed to accelerate an object with mass.* *relevant-property*
- *Metrics and requirements*: defining instruments for formulating the requirements of the system: the *metrics*. *Example: An average person can ride the bicycle with a speed of S for an hour without getting tired. A requirement defined in terms of this metric: the following should hold: $S \geq 12\text{km/h}$.* Several metrics may exist on different layers in the system extension hierarchy. The higher in the hierarchy, the more general the metric is (so, applicable to many, if not all system variants), the lower in the hierarchy the more limited the scope may be. Moreover, metrics on lower levels may be operationalisations of a metric on a higher level. *Example: High level: the practicability and comfortably of the bicycle; lower level: the aforementioned ‘average speed for an hour’-metric mentioned above.* Moreover, a metric on a lower level may represent an implementation of one or more metrics, in the sense that a requirement expressed in the latter is closer to suggesting ways to implement it than the first. *Example:* *metric requirement*
metric

High level: the aforementioned ‘practicability and comfortability’-metric; lower level: the stiffness of the material of which the frame of the bicycle is constructed, is a metric, because (in most cases) the stiffer the material, the less the energy expenditure, which implies lighter cycling. The latter metric suggest moreover how to implement a requirement: find a material with a certain stiffness. The first metric gives you ‘nothing to hold on’. It may be the case that there are different partially non-overlapping ways to operationalise higher level metrics or requirements.

development decision

- *Development decisions: providing potential development decisions and assess their performance using the aforementioned metrics. Example: The bicycle can be made of steel, carbon, wood, bamboo or another material. With respect to the layeredness of the design, the same thing as for metrics holds: the higher in the hierarchy, the more general the development decision.*

In addition to the elements, an iteration typically also provides examples to illustrate the effect of properties and/or development decisions on the quality. *Example: A bicycle made of bamboo has a different air resistance than a bicycle design made of steel, assuming that the rest of the design is as identical as possible.*

III.3.3.2 Notations

In the challenge-overview the individual assumptions each carry a label. The assumptions will be combined using the + sign, moreover, the assumptions form a ‘non-monotonic’ system: so a more restrictive assumption will get precedence over a less restrictive one. So, given $A + B$, if B is more restrictive, then for all states of affairs where B is applicable, assume B , for all the others, assume A . The abstractions get a label, too.

III.3.3.3 Tip of the iceberg

Some readers may think that the metrics that follow in this chapter may appear self-evident. However, the process of arriving at these metrics was everything but simple and self-evident. They were the result of quite some research effort. What is presented is the tip of an iceberg: I have conceived of and scrutinised a considerable number of metrics that did not make it. These metrics, however, inspired the formulation of the current ones. Even the current ones are not satisfactory in several respects and require further research.

III.3.4 Preliminaries

III.3.4.1 Overall metric: metric-for-integration-of-automated-deduction

In this section I propose the first version of the overall metric: the metric-for-integration-of-automated-deduction. First, I introduce two metrics that serve as its building blocks: the ancillary-metric-for-integration-of-automated-deduction and the metric-for-value-of-questions.

For the sake of simplicity, the rest of this chapter assumes that all metrics map into the same ordered set M . A possible choice is $M = [0, 1]$.

Definition 23 (ancillary-metric-for-integration-of-automated-deduction). *Given a human h , a community c of which h is part, a question q and a point of time t . Suppose that n is the answer that h would have arrived at around time t , would h have been able to evaluate each piece of information contained within c , and correctly apply human (and not automated) deductive reasoning to it, with utmost integrity and under the idealised assumption that this reasoning would take no time. ‘To evaluate’, here, is intended to have a similar meaning as ‘to take into account’, ‘to take notice of’ or ‘be aware of’. The information includes both information represented in some symbolic form on whatever medium, and ‘verbalisable’ thoughts of people. It also includes meta-information such as opinions about the level of expertise of individuals. The ancillary-metric-for-integration-of-automated-deduction $\text{intaudedu}^*(h, c, q, t)$ is the measure that quantitatively expresses the degree to which h is able to correctly apply automated-deductive-reasoners around time t to find answer n in a representation that is intelligible to h , and has confidence in this answer. It is expressed as a value from M . h is allowed to indirectly apply the automated-deductive-reasoners, for example, h may get assistance of someone else who is proficient in using reasoners, or the reasoners may be hidden as a ‘reasoning service’ under a computer interface.*

$\text{Dom}(\text{intaudedu}^*)$ is the notation for the domain of intaudedu^* , so for every human h , community c , time t and question q : $(h, c, q, t) \in \text{Dom}(\text{intaudedu}^*)$. An ancillary- intaudedu -function a is any function that maps $\text{Dom}(\text{intaudedu}^*)$ to M . ■

Part of the motivation of ancillary-metric-for-integration-of-automated-deduction follows now by reflecting on fragments of it:

- ‘around time t ’, ‘utmost integrity’, ‘the degree to which’, etc. These expressions are deliberately vague. Being well-defined has been sacrificed to achieve being complete. (Both notions were explained in section III.3.2.2.1 (p. 156).)

- ‘had h have been able to evaluate each piece of information’, ‘idealised assumption that this reasoning does not take time’. These expressions are not operational. Being operational has been sacrificed to achieve being complete.
- The word ‘information’ is a better word than ‘knowledge’. The word ‘knowledge’ implies that it is about things that are known to be true, which would exclude, among other things, uncertain facts, opinions and beliefs.
- ‘...utmost integrity’: Sometimes a person is highly biased. He or she may carry beliefs because of personal interests, in spite of overwhelming evidence that these beliefs are false. *Example: A highly addicted smoker may believe that smoking is healthy, because he cannot bear quitting, and also does not want to take the responsibility for his unhealthy choice.* Adding the word ‘integrity’ excludes these cases.
- ‘...and has confidence ...’: because having obtained the best answer does not necessarily imply that one has confidence in it. *Example: if the 5 year old daughter of a father from Miami states that the following statement is true: ‘The president of Nauri in 2017 was Baron Divavesi Waqa,’ he probably would consider this answer useless. While if he would read exactly the same information in Wikipedia he would have great confidence that the answer is correct.*
- ‘Suppose that n is the answer that h would have arrived, had h have been able to evaluate each piece of information contained within c ’: The metric has been defined such that it does not have to rely on a notion as ‘absolute truth’, which can never be attained in practice for most information about the world.
- ‘in a representation that is intelligible’: h could arrive at the exact answer n , however, in a representation that h does not understand. *Example: Akwasi does not know description logic, however the answer is presented in this formal-language.*

The ancillary-metric-for-integration-of-automated-deduction is not sufficient, because it does not capture the degree to which a human being values the answer to a given question. As will be shown later, capturing this notion as well is important to balance trade-offs while fostering integration-of-automated-deduction (section III.3.5.3 (p. 179)). A second ancillary metric is needed:

Definition 24 (metric-for-value-of-questions). *Let q be a question, and h be a human being, and t a point of time. $Qvalue(h, q, t)$ is a quantitative measure for how much h values finding an answer to question q around time t . It is expressed as a value from M .* ■

What remains to be done is to use the two building blocks, ancillary-metric-for-integration-of-automated-deduction and metric-for-value-of-questions to define a metric for the integration-of-automated-deduction into communities. There are different ways to do this, each emphasising other ways to, among other things, balance trade-offs. Therefore, the complete-metric will not be presented as a single metric but as a class of metrics⁴ The following defines the class. After that I present a concrete instance of the class, which I deem to be a viable metric-for-integration-of-automated-deduction.

Definition 25 (class-of-metrics-for-integration-of-automated-deduction). *A metric-for-integration-of-automated-deduction i is a function that maps an*

- *ancillary-metric-for-integration-of-automated-deduction* intaudedu* ,
- *a metric-for-value-of-questions* Qvalue ,
- *a community* c
- *and a point of time* t

to a degree that expresses the integration-of-automated-deduction into c around time t . This degree is expressed as a value from M . (Note that i is a higher-order function.)

The class-of-metrics-for-integration-of-automated-deduction is formed by the set of all metrics-for-integration-of-automated-deduction. ■

There are a number of conditions that should, reasonably, hold for any choice for a specific metric-for-integration-of-automated-deduction. To understand the following example of such a condition, it is important to distinguish a metric (e.g. the function intaudedu*) from its *actualisation*. This distinction can be illustrated by assuming there are multiple universes. The actual plot of function intaudedu* may look differently on different versions of the Earth. For example, imagine that on ‘Earth 1’, a brilliant computer scientist, Geoff Sutcliffe, brings researchers and developers in the field of automated-deduction together, accelerating the integration, while on ‘Earth 2’ he decided to become a professional sailor instead. In this case, intaudedu* may be increasing more steeply on Earth 1 than on Earth 2 for many humans. Another way to illustrate it is to compare the actualisations of intaudedu* from different existing communities, such as that of the Netherlands and that of Nigeria.

Example 1. *If i is a metric-for-integration-of-automated-deduction the following condition should hold. Let a_1 and a_2 be two different actualisations of intaudedu* . If for*

⁴Note that this situation is very similar to what is described in another context in this book, in section II.5.6.3 (p. 82).

all $x \in \text{Dom}(\text{intaudedu}^*)$ it holds that $a_1(x) < a_2(x)$ then $i(a_1) < i(a_2)$. Informally speaking: if a_1 is smaller than a_2 in all points, then the score on metric-for-integration-of-automated-deduction of the first has to be smaller as well. Note that in this case the question-value-function does not matter. ■

Extending this list of conditions is left as potential future work.

The following defines a specific metric-for-integration-of-automated-deduction. It is based on averaging over all humans in the given community and all questions, weighted by the value each of these humans assign to each question.

Definition 26 (metric-for-integration-of-automated-deduction- α). Assume that the set of all possible questions Q is indexed, so $Q = \{q_0, q_1, q_2, \dots\}$. Metric-for-integration-of-automated-deduction- α (intaudedu_α) is the function that maps an actualisation of ancillary-metric-for-integration-of-automated-deduction a , an actualisation of metric-for-value-of-questions v , a community c and a point in time t as follows:

$$\text{intaudedu}_\alpha(a, v, c, t) = \frac{\sum_{h \in c} \lim_{n \rightarrow \infty} \left(\frac{\sum_{i=0}^n a(h, c, t, q_i) \times v(h, t, q_i)}{\sum_{i=0}^n v(h, t, q_i)} \right)}{|c|} \quad (\text{III.3.1})$$

Note that definition 26 uses a limit of $n \rightarrow \infty$. Whether the behaviour of the above definition is free of undesired effects when there would be infinitely many questions with a value different from zero, is not yet clear. However, I believe this is not a problem. I believe that it is reasonable to assume that there are, at any moment in time, only finitely many questions that are valued by any human to be answered. This means that the value of the metric is always determined in a finite number of steps.

III.3.4.1.1 Customising the metric-for-integration-of-automated-deduction

The metric-for-integration-of-automated-deduction- α emphasises an integration-of-automated-deduction into a community, in which the average member of that community scores well. Of course, this is not the only viable choice because of various trade-offs. For example, perhaps in a certain time, it may be important to give a certain subcommunity priority. A concrete example would be if we would want to prevent the Earth from total annihilation by an impending asteroid impact. In that case, we may want to assign a higher weight to the questions posed by the people who are working on deflecting the asteroid's trajectory. Therefore, I believe that the choice for instances of the class-of-metrics-for-integration-of-automated-deduction should be tailored to the circumstances of a specific community.

III.3.4.1.2 Time

The metrics-for-integration-of-automated-deduction contains a time dimension. It expresses integration-of-automated-deduction as a function of time. Of course, the ideal is a perfect score at any time t . However, in practice this will never be achieved, and among other things, trade-offs will manifest themselves. Therefore, it may be useful to define an additional metric for integration-of-automated-deduction on an even higher level by assigning a measure to such functions of time. In other words, this higher-level metric maps a development of integration-of-automated-deduction in time to a measure in M .

Depending on the context of the actors fostering integration-of-automated-deduction they may pursue a different development of integration-of-automated-deduction in time. For example, the actors can invest in short-term gains at the cost of long-term gains. This could, for example, be the case if actors choose to dedicate resources to express new information in KR-bases (short-term gains), instead of dedicating them to developing good education in how to use reasoners and developing better reasoners (long-term gains).⁵

III.3.4.1.3 Operational metrics

The metrics-for-integration-of-automated-deduction are neither operational nor well-defined. The definition of a suite of metrics with these properties is desirable for actual assessment of the state of affairs concerning the integration-of-automated-deduction. The development of such metrics is left as future work. Some preliminary thoughts about this topic, however, follow now.

First, a candidate could be based on statistical sampling. Given a community, draw a representative sample of humans of that community, and then ask each of these to, for a week, maintain a list of questions that are valuable for them to be answered. For each question they have to indicate the degree they value the answer. After that week, ask them to apply automated-deductive-reasoners to deduce the answers.

Note that the metric could be regarded as a new type of metric, for which I coin the term *approximate-metric*. This is a metric that approximates another metric, and would theoretically converge to this metric if all information were available.

*approximate-
metric*

The metric just sketched is unfortunately still not fully operational: how can one be certain that the test-person indeed applied reasoners and did so correctly? Much automated-reasoning activity might, for example, take place under the hood of applications, perhaps run on inaccessible proprietary software on proprietary

⁵This situation is very similar to what is described in another context in this book, in section II.5.6.3 (p. 82).

KR-bases. This situation may be improved by sacrificing completeness, and setting up a competitive setting. This is comparable to the World Solar Challenge. For example: select a community and a domain of interest, and give this community two months to train and prepare, including filling an open source KR-base in languages used by open source automated-deductive-reasoners. An example of a community could be a group of 10 students in sailing and the domain of interest could be sailing. Additionally, ask another group of people who are active in the given domain to keep a list of questions regarding the domain that are valuable to them. After this period the first group has to try to deduce the answers to the questions by applying automated-deductive-reasoners. This metric, with some further details filled in, is operational. Obviously, though, it is not complete.

III.3.4.2 The choice list of relevant assumptions

The assumptions used in the following sections are provided in the following ‘choice list’. As stated before, each of these sections may apply different subsets of these assumptions to isolate certain development challenges. Therefore the following list is *not* to be interpreted as all relevant-assumptions that hold throughout this section.

Assumption 1 (formal-super-language). There exists a formal-language, *formal-super-language*, in which all knowledge can be expressed in such a way that it allows maximum deducability. *formal-super-language*

Assumption 2 (ideal-translator-oracle). There is an ever-existing oracle, *ideal-translator-oracle* that translates any new occurrence of information in a given community instantly (in zero time) into the formal-super-language (as defined in assumption 1). The information may occur in any form, be it written in natural language, in a formal-language or in the thoughts of people. There is one aspect of the translation that may not be ideal. Pieces of information can be translated in different semantically equivalent expressions. Depending on a given reasoner, this may, among other things, influence the time it may take this reasoner to reach certain reasoning-goals. The ideal-translator-oracle, however, translates pieces of information always into the same expressions. *ideal-translator-oracle*

Assumption 3 (ideal-reasoner-oracle). There is an ever-existing oracle, *ideal-reasoner-oracle* that can reach any (computable) reasoning-goal in zero time (so, it also tacitly assumes to have an unlimited amount of memory available, which can be accessed in zero time). *ideal-reasoner-oracle*

Assumption 4 (one-user). There is only one entity (for example, a single human user) that benefits from integration-of-automated-deduction, so there is no conflict of interests, also it is assumed that this one user does not have an internal conflict of interests or preferences.

Assumption 5 (consistent-information). All information in the community is at all times free from inconsistencies. Note that it is, therefore, also free from ambiguities as far this is needed to prevent these inconsistencies. E.g. “Akwasi is at the organic food market” and “Akwasi is at the basketball game.” are inconsistent, unless you add “Akwasi is at the organic food market at December the 9th, 2014.” and “Akwasi is at the basketball game at December the 13th, 2014.”

Assumption 6 (perfect-reasoner-accessibility). At all times, all people in the community have access to all automated-deductive-reasoners. ‘Access’ in this context means that they can directly or indirectly apply these automated-deductive-reasoners at any moment.

Assumption 7 (varying-importance-of-information). The importance of a translation of a specific fragment of natural language expressions may vary depending on the user and the context of this user.

Assumption 8 (time-consuming-translator-oracle). There is an ever-existing oracle, *time-consuming-reasoner-oracle* that translates any new piece of information that *time-consuming-reasoner-oracle*

occurs in a community into formal-super-language. However, in contrast to ideal-translator-oracle it does so in a process that takes time. Time-consuming-reasoner-oracle still has an unlimited amount of memory at its disposal. It follows some strategy to determine in which order the translation takes place.

Assumption 9 (fixed-reasoners-per-formal-language). For each formal-language there is a limited and fixed set of automated-deductive-reasoners available.

Assumption 10 (one-formal-language). There is only one formal-language available.

Assumption 11 (unique-reasoner-per-formal-language). For each formal-language in use, there exists a unique automated-deductive-reasoner.

Assumption 12 (fixed-set-of-formal-languages). There is a predefined set of formal-languages C . Everything expressed in a formal-language, is expressed in one of the languages from C . This limitation is an abstraction of the fact that in reality it is not feasible to translate texts into all languages, for several reasons.

Assumption 13 (ever-extensible-formal-language-set). The aggregate of available formal-languages can be extended indefinitely. This is a realistic assumption.

Assumption 14 (full-intelligibility). All statements deduced with the existing automated-deductive-reasoners from existing KR-bases are intelligible to each human.

Assumption 15 (reasoner-chooser-oracle). There is an ever-existing *reasoner-chooser-oracle* that chooses the automated-deductive-reasoner that will perform best for a given reasoning-goal.

III.3.5 Focus-based challenge analysis

III.3.5.1 Fostering integrations of automated deduction under Ideal assumptions

The focus-based challenge analysis starts with what could be considered the top-level 'naive' assumptions, **Ideal**. These are assumptions that may be made tacitly by laymen. These assumptions function as a conceptual aid to present the basic system design, which is also present in the subsequent design-refinements. This approach has been explained in section III.3.3 (p. 162). It appears that under these assumptions, there are no challenges in fostering integration-of-automated-deduction.

III.3.5.1.1 Relevant assumptions

All assumptions are formulated with respect to a given community of people.

$$\begin{aligned} \mathbf{Ideal} &= \text{ideal} \cdot \text{translator} \cdot \text{oracle} + \text{ideal} \cdot \text{reasoner} \cdot \text{oracle} \\ &+ \text{consistent-information} + \text{perfect-reasoner-accessibility} + \text{full-intelligibility} \end{aligned} \quad (\text{III.3.2})$$

III.3.5.1.2 Challenges

There are no development-challenges, at least not when applying the metric-for-integration-of-automated-deduction. It will always have a perfect score according to this metric.

III.3.5.1.3 Potential development decisions

The metric-for-integration-of-automated-deduction is sufficient, and, this universe always achieves a perfect score on this metric (= the amount of improvement by indirect or direct application of automated-deductive-reasoners realises its maximum potential in this case), for any question q , human h and point of time t . Explanations are as follows:

1. Because of perfect-reasoner-accessibility, everyone has access to the ideal-reasoner-oracle,
2. Because of the fact that the ideal-translator-oracle translates everything immediately into formal-super-language, everything that can be deduced from it, is instantly accessible to h , including the answers to h 's question if they are deducible.
3. Because of consistent-information the following problem cannot occur. Suppose that the information in the world would contain inconsistencies. Then these inconsistencies can be established immediately by the ideal-reasoner-oracle, and as such help h . However, after that h has to deal with these consistencies in one way or the other, for example by selecting consistent subsets he thinks are better than others. The latter would imply that he does not necessarily reach the answer that he would have arrived at if he would have been able to evaluate each piece of information (where 'evaluate' is used as in the sense described in definition 23). This also depends, among other things, on h 's capabilities and tools to deal with these inconsistencies, such as

finding out which sources of information he trusts. Fortunately, all information is consistent, so this problem cannot occur.

4. Because of full intelligibility, the deduced representation will be intelligible.

Hence, with this assumption set, a perfect score for metrics for integration of automated deduction is achieved. This also implies there are no challenges in this case.

III.3.5.2 Metric: automatically deducible fraction

In this section I propose a sub-metric of the metric for integration of automated deduction. It is a metric that becomes relevant as soon as the ideal translator oracle assumption does not hold, which is, of course one of the most unrealistic assumptions listed in section III.3.4.2 (p. 170). A central aspect of the metric for integration of automated deduction is formed by the creating representations of information in a formal languages such that automated deductive reasoners can deduce as much as possible from these representations. This description, however, is vague. How should one quantify the latter notion? What is 'as much as possible'? If one compares two representations, which should be considered to be better in this respect?

The proposed metric is predominantly well-defined. The metric, however, will in most cases not be operational.

III.3.5.2.1 Strategy

The basic idea is to define a metric for the 'deductive' quality of a formalisation F of a given body of information B sketchwise as follows. First, we limit B to all information that is expressed or expressible in some symbolic language, whether it be natural language or a formal language. It is assumed that F exclusively expresses states of affairs that are also expressed in B . Determine the set of sentences that can be deduced from F . Also determine the set of sentences that can be deduced from B . Evidently, the last occurrence of the word 'deduced' refers to deduction in the general sense, and not only the formal sense. After all, B is possibly expressed in natural language. The basic idea is now to define a metric that is inspired by the determining the ratio between the sizes of the latter two sets.

Merely taking the ratio does not produce an adequate metric. There are a few obstacles that have to be overcome. For one, the set of sentences that can be deduced from F is infinitely large, because there are infinitely many tautologies. The same holds for B . *Example: In propositional calculus, all sentences in the infinite set $\{p \vee \neg p, (p \wedge p) \vee \neg p, (p \wedge (p \wedge p)) \vee \neg p, \dots\}$ are true, whether p is false or true.* This means that the mentioned ratio is undefined, because it is a division between infinities. In

this section, I propose to solve this problem, by first defining a local, or bounded, form of deduction, then define a local variant of the ratio, and then take the limit of this local version to infinity. In the proposed implementation, the locality is realised by limiting the proofs (of B) to a maximum length. This means that the set of sentences are limited to those that can be deduced with proofs with at most this length. There is an intuition behind this choice for boundedness. Limitations in computational resources limit the maximal length of proofs that can be produced by automated-deductive-reasoners. Hence, it makes sense to ‘walk the path to infinity’ from the sentences that are deducible the easiest to those that are further away.

A second challenge is that the calculation of the ratio requires information expressed in different languages to be compared. When is a sentence in predicate logic equivalent to one in natural language, in propositional calculus or in description logic? For this I propose a way to structurally compare sentences of different languages.

III.3.5.2.2 Structurally comparing languages

In order to compare the same information expressed in distinct languages with respect to their deductive behaviour, it is important to establish which sentence in one language forms a ‘direct’ translation of a sentence in another language. The languages referred to include formal-languages and natural languages, or any other language that expresses information using sequences of symbols. For this I introduce the notion symbolic-language, which is any language that expresses information as a sequence of symbols over a finite alphabet. It includes natural languages and formal-languages.

This is not a trivial matter. Among other things, it is not sufficient to state that two sentences must express the same information. For example, the sentence “Akwasi wears a red hat.” and “If Fermat’s last theorem is true, then Akwasi wears a red hat.” extensionally have the same meaning, since Fermat’s last theorem is true. However, they express the information in a completely different way. Assessing whether the second sentence expresses that Akwasi indeed wears a red hat, requires a tremendously long proof (Wiles, 1995). For this reason, one also needs to somehow compare the sentences with regard to their *structure*. This is the purpose the following definitions.

Definition 27 (structural-sentence-compatibility). *Let \mathcal{L}^*_1 and \mathcal{L}^*_2 each be a symbolic-language. Let φ^*_1 be a sentence in \mathcal{L}^*_1 and φ^*_2 a sentence in \mathcal{L}^*_2 . Then the sentences φ^*_1 and φ^*_2 are structurally-compatible if φ^*_2 can be obtained by substituting substructures of φ^*_1 with substructures of φ^*_2 that could semantically express the same information. Each substitution may be used only once.* ■

This definition is non-formal, because it includes non-formal-languages. Moreover,

the substitution referred to, is intended to be quite ‘forgiving’. Minor syntactical differences are ‘forgiven’. I do not know whether it is possible to define a strict non-forgiving substitution in all cases. The ‘forgiving’ definition is sufficient to serve the purpose of this work.

Example 2.

1. Let $\varphi_1^* = p_0 \rightarrow p_1$ (propositional calculus) and $\varphi_2^* = (\exists x.M(x)) \rightarrow (\exists x.P(x))$ (predicate logic sentence). These sentences are structurally-compatible, because φ_2^* can be obtained from φ_1^* by the following substitutions: replace p_0 with $(\exists x.M(x))$ and p_1 with $(\exists x.P(x))$. More intuitively expressed: the implication is expressed in both, φ_2^* , however, contains more details.
2. The sentence $\forall x : (M(x) \rightarrow P(x))$ (predicate logic) is structurally-compatible with “if an individual is a man, then that individual wears a purple hat” (natural language). Substitutions: $M(x)$ with “an individual is a man”, $P(x)$ with “that individual wears a purple hat” and $\forall x : (\varphi_1(x) \rightarrow \varphi_2(x))$ with “if $\text{Trans}(\varphi_1(x))$, then $\text{Trans}(\varphi_2(x))$ ”, where Trans is a function that applies substitutions that are provided.⁶

■

Definition 28 (sentence-equivalence (=*)). Let \mathcal{L}^*_1 and \mathcal{L}^*_2 be two languages. These can be any symbolic language, including natural languages and formal-languages. Let φ_1^* be a sentence in \mathcal{L}^*_1 and φ_2^* a sentence in \mathcal{L}^*_2 . Then sentences φ_1^* and φ_2^* are equivalent if they are structurally-compatible and if they express the same state of affairs. It is written as $\varphi_1^* =^* \varphi_2^*$

■

Example 3.

1. Let $\varphi_1^* = p_0 \rightarrow p_1$ (propositional calculus) and $\varphi_2^* = (\exists x : M(x)) \rightarrow (\exists x : P(x))$ (predicate logic). In example 2 it has been shown that these sentences are structurally-compatible. They also express the same state of affairs if the atomic expressions, from the perspective of the given substitutions, of both languages are ‘grounded’ in the same way. In this case, if p_0 expresses the same state of affairs as $(\exists x : M(x))$ and p_1 the same as $(\exists x.P(x))$.
2. The sentences $\forall x : (M(x) \rightarrow P(x))$ (predicate logic) and “if an individual is a man, then that individual wears a purple hat” (natural language) are structurally-compatible. This has been shown in example 2. They also express the same state of affairs if $M(x)$ means “ x is a man”, $P(x)$ means “ x wears a purple hat”.

■

⁶The substitutions are given sketch-wise. Fully dealing with quantifiers and variable bindings is more complicated.

III.3.5.2.3 Local deductive closure

The strategy in section III.3.5.2.1 is based on a local version of the common notion deductive-closure. The following first provides the definition of deductive-closure.

Definition 29 (deductive-closure). *Given are a formal-language \mathcal{L} including a particular proof system S . Let Γ be a set of sentences in \mathcal{L} . Then the deductive-closure is the set of all sentences that are deducible from Γ . In this book, it is written as $\text{clo}(\Gamma)$.* ■

For the local version of the deductive-closure, the notion proof-length is needed.

Definition 30 (proof-length). *Given is a formal-language \mathcal{L} including a particular proof system S . Let d be a proof in S . Then this proof can be written as a sequence of symbols over a finite alphabet. Assume that there is a standard way of doing so for the given proof system. The proof-length of d is the number of symbols in d if it is written as a sequence of symbols. It is written as $|d|$.* ■

Note that Pudlák (1998) provides the above and alternative definitions of proof-length. He refers to the above definition as the ‘proof size’. For the purposes of this work, this definition suffices.

Definition 31 (local-deductive-closure). *Given are a formal-language \mathcal{L} including a particular proof system S . Let Γ be a set of sentences in \mathcal{L} . Then the local-deductive-closure is the set of all sentences that are deducible from Γ with a proof-length of at most n . It is written as $\text{loclo}(\Gamma, n)$ (local deductive closure).* ■

Definition 32 (general-local-deductive-closure). *The general-local-deductive-closure is a generalisation of the local-deductive-closure. It is generalised to include non-formal-languages as well, in particular natural language. It does so by generalising the notion of proof to include any deductive proof. The generalised notion includes non-formalised proofs (i.e. proofs that are not stated in a formal proof system), such as proofs stated in natural language, or proofs as they are normally written in mathematics. It assumes that there exists a knowledgeable jury, who can establish whether the proof is indeed a sound deductive proof, deducing the given conclusion from the given premises. This definition, is therefore itself also non-formal. It is written as $\text{loclo}^*(\Gamma^*, n)$, where Γ^* is a set of sentences in any mix of symbolic-languages, and $n \in \mathbb{N}$. It is limited to sentences in the languages that occur in Γ^* .* ■

Note that the limitation to sentences phrased in the symbolic-languages that occur in Γ^* is important. Theoretically it is possible that there are infinitely many natural languages. In that case, the general-local-deductive-closure would always be infinitely large. Each sentence, and all its infinite equivalent translations would

occur in the closure. *Example:* ‘Akwası is a man.’, ‘Akwası ist ein Mann.’, ‘Akwası est un homme.’, ‘Akwası bu nwoke.’, and so forth for each of the infinitely many natural languages.

This problem, however, can also be solved without resorting to this limitation, and perhaps in an even more elegant way. To wit: by partitioning the local-deductive-closure by means of equivalence classes induced by sentence-equivalence. *Example:* In the previous example with ‘Akwası is a man.’ and so forth, all the mentioned examples would be united in a single set, and thus count as one item.

III.3.5.2.4 Automatically-deducible-fraction

A few auxiliary definitions:

Definition 33 (equivalent-in (\in^*)). *Let Γ^* be a set of sentences in any mix of symbolic-languages. Let φ_1^* be a sentence formulated in one of these symbolic-languages. Then equivalent-in (\in^*) is defined as follows:*

$$\varphi_1^* \in^* \Gamma^* \Leftrightarrow \exists \varphi_2^* : \varphi_2^* \in \Gamma^* \text{ and } \varphi_2^* =^* \varphi_1^*$$

(Note that $=^*$ has been defined in definition 28.) ■

Definition 34 (intersect-under-equivalence (\cap^*)). *Let Γ_1^* and Γ_2^* each be a set of sentences in any mix of symbolic-languages. Then the intersect-under-equivalence is defined as follows:*

$$\Gamma_1^* \cap^* \Gamma_2^* = \{\varphi^* \mid \varphi^* \in^* \Gamma_1^* \text{ and } \varphi^* \in^* \Gamma_2^* \text{ and } \varphi^* \in \Gamma_1^* \cup \Gamma_2^*\}$$

The last condition is added to limit the sentences to those from the given sets. Otherwise, the intersection would contain any possible equivalent sentence in any language. ■

This almost concludes the formalisation of the strategy described in section III.3.5.2.1 (p. 174). One preparatory definition remains:

Definition 35 (local-automatically-deducible-fraction). *Let Γ^* be a set of sentences in any mix of symbolic-languages. Let Γ be the subset of sentences of Γ^* in a formal-language. The local-automatically-deducible-fraction is defined as:*

$$\text{local_deducible_fraction}(\Gamma, \Gamma^*, n) = \frac{|\text{locl}^*(\Gamma^*, n) \cap^* \text{clo}(\Gamma)|}{|\text{locl}^*(\Gamma^*, n)|}$$

■

The pursued metric is then obtained by taking the limit to infinity of the local-automatically-deducible-fraction:

Definition 36 (automatically-deducible-fraction). *Let Γ^* be a set of sentences in any mix of symbolic-languages. Let Γ be the subset of sentences of Γ^* in a formal-language. The automatically-deducible-fraction is defined as:*

$$\text{deducible_fraction}(\Gamma, \Gamma^*) = \lim_{n \rightarrow \infty} \text{local_deducible_fraction}(\Gamma, \Gamma^*, n)$$

■

III.3.5.3 Translation prioritisation

III.3.5.3.1 Relevant-assumptions

TranslationPrioritisation =

Ideal

–ideal-translator-oracle + time-consuming-translator-oracle

+varying-importance-of-information + one-user

(III.3.3)

The core of the change is formed by the replacement of the ideal-translator-oracle by the more realistic time-consuming-translator-oracle.

III.3.5.3.2 Relevant properties

Without the ideal-translator-oracle, translations will take resources (time and space, etc.) to be carried out.

III.3.5.3.3 Metrics and requirements

I propose to extend the suite of metrics with one based on the automatically-deducible-fraction. The reason is that the order of translation is now important because of varying-importance-of-information. Some information may be more important at a certain time to be translated.

Definition 37 (weighted-local-automatically-deducible-fraction and weighted-automatically-deducible-fraction). *The weighted-local-automatically-deducible-fraction and the weighted-automatically-deducible-fraction are respectively a variation on the local-automatically-deducible-fraction and the automatically-deducible-fraction (see definition 35 on the facing page and definition 36). In these, the size of the sets in the definitions is replaced by a weighted sum. In the summation, each sentence is weighted by its degree of importance.*

■

The weighted-automatically-deducible-fraction is directly related to the metric-for-value-of-questions (see definition 24 (p. 166), definition 25 (p. 167) and definition 26

(p. 168)). A sensible choice in the weighted-automatically-deducible-fraction is to assign to each deducible sentence the average value of the question it answers at a given point of time.

The weighted-automatically-deducible-fraction is better than the automatically-deducible-fraction, because it is influenced by the order in which the time-consuming-reasoner-oracle translates the information.

III.3.5.3.4 Challenges

The challenge is to determine a good strategy to determine what to translate first, as to optimise weighted-automatically-deducible-fraction.

III.3.5.4 Trade-off expressivity and complexity

III.3.5.4.1 Relevant assumptions

In a scheme, the assumption-set is as follows:

$$\begin{aligned} \text{ExpressivityVsComplexity} = & \\ \text{Ideal} & \\ -\text{formal-super-language} + \text{ever-extensible-formal-language-set} & \quad (\text{III.3.4}) \\ -\text{ideal-reasoner-oracle} & \end{aligned}$$

III.3.5.4.2 Relevant properties

III.3.5.4.2.1 Trade-off In the 1980s it was discovered, or at least came to the awareness of the knowledge representation and reasoning research community that there is a trade-off between the expressivity and complexity of formal-languages. From a certain level of expressivity you even lose decidability. These days, a part of the community therefore added an additional requirement concerning the decidability of the formal-language, see below.

III.3.5.4.2.2 Existence of a decidable ‘sufficiently expressive’ formal-language unknown First a definition:

Definition 38 (sufficiently-expressive-for-human-usage). *A formal-language \mathcal{L} is sufficiently-expressive-for-human-usage if any ordinary human user of automated-deductive-reasoners, now, or in the future, can use it to express any body of information and any question for which it holds that (1) there exists a formal-language in which these are expressible, and (2) they are useful for him or her to express.* ■

The ordinary human user spoken of is someone who wants to apply integration-of-automated-deduction to deal with real-world problems. So, this context, among others, excludes logicians or philosophers of logic who are investigating logical systems. It may be clear that for the latter, there is no limit to what expressivity they need for their discourse. Among other things, investigations into expressivity itself are at the core of their activities.

Note that this definition is not a formal definition. It is probably impossible to formalise this notion. The problem lies in the notion ‘useful to any human being’, which is subjective.

A formal-language sufficiently-expressive-for-human-usage is ideal for human usage with regard to its expressivity: more expressivity is not needed. To the best of my knowledge, there are no decidable languages currently known to be sufficiently-expressive-for-human-usage. This does not exclude the possibility that they exist but that we simply have not found them yet.

The opposite can also be true. Such languages may not exist. It is known that there are formal-languages that are undecidable, and have a greater expressivity than any decidable formal-language. Hence, if there are ordinary users who need such expressivity, then there are no decidable languages that are sufficiently-expressive-for-human-usage.

III.3.5.4.2.3 Trade-off between language constructs in decidable formal-languages It turns out there is a trade-off among some sets of formal-language constructs, if you want to remain within the boundaries of decidability. This is introduced with some more precision in the following.

Definition 39. *A formal-language \mathcal{L}_1 is as expressive as \mathcal{L}_2 if everything that can be expressed in \mathcal{L}_1 , can also be expressed in \mathcal{L}_2 , and vice versa.* ■

Definition 40. *A formal-language \mathcal{L}_1 is more expressive than \mathcal{L}_2 if everything that can be expressed in \mathcal{L}_2 , can also be expressed in \mathcal{L}_1 , but even more than that.* ■

Definition 41. *A formal-language \mathcal{L}_1 and \mathcal{L}_2 have partially overlapping expressivity, if there are states of affairs that can be expressed in \mathcal{L}_1 but not in \mathcal{L}_2 , and vice versa.* ■

Theorem 1. *There exist decidable languages \mathcal{L}_1 and \mathcal{L}_2 with partially overlapping expressivity, such that any language \mathcal{L}_3 which is at least as expressive as \mathcal{L}_1 and \mathcal{L}_2 , is undecidable.*

Proof. This is proven by providing an example: choose the following description logic languages (Baader, 2003): $\mathcal{L}_1 = \mathcal{ALC}(\cap, \cup, \neg(full), *)$ and $\mathcal{L}_2 = \mathcal{ALC}(\cap, \cup, \circ, *)$, and $\mathcal{L}_3 = \mathcal{ALC}(\cap, \cup, \circ, \neg(full), *)$. (Where: \mathcal{ALC} is the attributive language with

complements, \cap is role intersection, \cup is role union, \circ is role chain (composition), \neg (full) is full role complement and $*$ is reflexive transitive closure.) Moreover, choose as the reasoning problem ‘concept satisfiability’. This latter problem is: decidable for \mathcal{L}_1 (Fischer and Ladner, 1979), decible for \mathcal{L}_2 (Lange and Lutz, 2005), but undecidable for \mathcal{L}_3 (Schild, 1989). Moreover, \mathcal{L}_3 is the least expressive language that is at least as expressive as \mathcal{L}_1 and \mathcal{L}_2 as becomes clear by simply reading their definition. \square

The latter demonstrates that there exist pairs of language constructs that, when added both to a language, can mutually interact such that they produce undecidability, while adding one of them, does not do so. This confirms the existence of the aforementioned trade-off. Note that from this it also follows there is no maximally expressive decidable formal-language in the sense that it can express what any other decidable formal-language can express.

III.3.5.4.2.4 Expressivity ordering according to human usage The previous relation, however, is not the whole story. The goal is to establish a formal-language for *human usage*, not the most expressive formal-language. The following definition is intended to capture this goal.

Definition 42 (better-expressivity-for-human-usage). *A formal-language \mathcal{L}_1 has a better-expressivity-for-human-usage than formal-language \mathcal{L}_2 , if the set of states of affairs and questions about these that can be expressed with \mathcal{L}_1 and that are useful to at least one human, in the past, present or future, is a superset of the set of such states of affairs expressible with \mathcal{L}_2 .* \blacksquare

Note that this definition is not about convenience of syntax. It is about semantics, so being capable of expressing a certain state of affairs, whether it be convenient given the syntax of the language or not. It is important to realise that even if \mathcal{L}_1 is more expressive than \mathcal{L}_2 , the previous definitions do not demand that \mathcal{L}_1 has a better-expressivity-for-human-usage. That is exactly the case if the added expressivity does not increase the set of states of affairs useful for humans to be expressed.

III.3.5.4.2.5 Trade-off between language constructs in decidable formal-languages from the human perspective The previous observations produce a perhaps confusing fact. In spite of trade-offs between language constructs in decidable formal-languages, it is still possible that there exists a decidable formal-language which is sufficiently-expressive-for-human-usage. If this is the case, the trade-off would simply not matter. In other words the language constructs added that make the formal-language undecidable can perhaps be replaced by other constructs that bring the language within the boundaries of decidability, while providing exactly sufficient

expression possibilities for human usage. However, to the best of my knowledge we do not know whether such a formal-language exists. Lets conclude with summarising this in the form of an hypothesis:

Hypothesis 1. *There does not exist a formal-language that is decidable and sufficiently-expressive-for-human-usage.*

III.3.5.4.3 Metrics and requirements

There are theoretical lower bounds to the complexity of the automated-deductive-reasoners that one can construct for a given formal-language.

Definition 43 (known-formal-language-complexity). *A known lower bound of the complexity of automated-deductive-reasoners that can be constructed for this language.* ■

‘Known’ is added, because there are cases in which it is not yet known whether the lower bound is the lowest one. For example, as long as the long standing theorem $P \neq NP$ has not been proven, it may be so that a formal-language with known lower bound NP , has a sharper lower bound in P .

The following metric covers the case that a certain lower bound has been proven to be the lowest:

Definition 44 (formal-language-complexity). *The formal-language-complexity is the complexity of automated-deductive-reasoners that has the lowest complexity that can be constructed for the given formal-language. For example, the formal-language may have been proven to be undecidable, which means that there do not exist automated-deductive-reasoners that can deduce the answer to all queries, given a KR-base.* ■

Note that formal-language-complexity and known-formal-language-complexity will in general not be given as a sharp bound, but rather as a set of complexities with a certain ‘bandwidth’. E.g. P is not a specific complexity, but a complexity *class*. For example, if formal-language-complexity is known to be undecidable, there is still a lot of differentiation that can be made, because for some formal-languages there may exist algorithms that perform better on the inputs for which they terminate than other formal-languages.

III.3.5.4.4 Potential development decisions

A considerable part of the knowledge representation and reasoning research community requires the formal-language to be at the least decidable. In that case, the user can be certain that (s)he will always get an answer to the questions asked – if we

assume that the reasoning times will not be too long. In particular, the description logic research community has made it into an art to explore the trade-offs between expressivity and complexity (Baader, 2003).

In general, the known-formal-language-complexity can be used as a guideline to press down complexity as much as possible, while still being able to express (most) states of affairs that a human user wants to express. If hypothesis 1 on the previous page is indeed true, then this is an ever-ongoing process. In other words, the choice for a formal-language depends on what a human user wants to express, and what expressivity he is willing to sacrifice for the sake of decidability. However, even if hypothesis 1 on the preceding page is false, a similar process may take place on a 'lower' level. There may be trade-offs between complexities of decidable formal-languages as well. The question then becomes what expressivity the user is willing to sacrifice to have a formal-language which is in a lower complexity class.

III.3.5.5 Fragmentation of formal-languages

One of the problems one encounters while optimising the degree of integration-of-automated-deduction (according to the metric-for-integration-of-automated-deduction) is fragmentation within the used formal-languages. This is best explained with a few examples (it is a running example). The examples, among other things assume that the reader is somewhat familiar with description logic (Krötzsch et al., 2014; Baader, 2003) .

An example of fragmentation in the information expressed with a single formal-language follows.

Example 4. *Given a KR-base in FOL that consists of the following set of sentences: $\Gamma = \{\forall x : (P_0(x) \wedge P_1(x)) \rightarrow Q(x), P_0(c_0), P_1(c_1)\}$. Also suppose that constants c_0 and c_1 refer to the same entity (without this equality having been expressed in the KR-base). A reason for this could be that constants c_0 and c_1 were added to the KR-base by two independent persons. Then we cannot make the inference: $\Gamma \vdash Q(c_0)$, while it is true. ■*

An example of fragmentation caused by expressing information in different formal-languages is as follows.

Example 5. *We add FOL statement $\exists x : P_0(x)$ to the KR-base, and the description logic statement $P_1 \equiv \top$ (meaning: P_1 has the same extension as the universal concept top, i.e. P_1 holds for all individuals of the universe). Moreover, P_1 in the latter corresponds with the predicate P_1 in the FOL statements. An automated-deductive-reasoner constructed for FOL cannot do anything with the second statement, so it cannot infer that $\exists x.Q(x)$. A reasoner for description logic cannot use the first statement, and can therefore also not make the given inference. ■*

Note that using different formal-languages does not necessarily lead to fragmentation. There is no fragmentation if there exists an automated-deductive-reasoner that is capable of coping with the given formal-languages in an integrated way, in conjunction with the addition of some equivalences between information expressed in both languages.

Fragmentation within the formal-languages reduces the scores on metrics-for-integration-of-automated-deduction. The two ways to reduce the fragmentation are: (1) standardisation of languages, and (2), definition of semantic equivalences in a formal-language.

A further elaboration on standardisation of languages follows now. Standardisation of language means the following:

- All semantic conditions are known/shared (or easily accessible) by all members of the community. (Note that this does not require a unification of all languages into one language.)
- The complete body of semantic conditions is free from inconsistencies. Hence, a given (complete) expression has an unambiguous interpretation. An example violating this rule is: if, in the context of the user, $P(c)$ has an interpretation as in standard predicate logic, but also has an other interpretation, for example one in which the roles of predicate and constant are reversed: c being the predicate and P the constant.
- Machines constructed that use or create formal-language expressions exploit all existing (relevant) semantic conditions for their purpose. (Examples of machines that could employ formal-languages are temperature sensors of weather forecasting services and satellites in orbit gathering valuable information.)

Further insight in standardisation of languages is achieved by the following observation I made concerning formal-languages' semantic conditions. I believe they can be categorised into two distinct types:

1. The *structural-semantic-conditions*: these are the conditions that are commonly associated with symbolic logic systems (KR-languages, symbolic logic, logic languages). I call them *structural* because symbolic logics' semantics are defined in terms of (mathematical) structures, in which one abstracts from every other association. *structural-semantic-conditions*
2. The *domain-related-semantic-conditions*: these are the conditions that come into play, next to the structural semantic conditions, if we apply a logic to make statements about something other than mathematical structures. An example is the semantic condition that constant b corresponds with the Nigerian novelist *domain-related-semantic-conditions*

Ben Okri. Mostly, natural language expressions that already have a semantics are used to define these semantic conditions.

This implies that using a shared language constitutes:

- Using a shared set of structural semantic conditions. This is exactly what happens when people decide to use the same ‘logic’ or ‘knowledge representation language’, for example RDF, OWL, Prolog, Relational Databases, etc.
- Using a shared set of domain related semantic conditions, in particular the interpretation of identifiers for entities (‘terminology’). In the literature, the latter are often called ‘ontologies’ (Gruber, 2009), and alternative ways of formulating the process is: creating a shared ontology.

Note that ‘sharing semantic conditions’ does not only refer to sharing semantics, but also to sharing syntax. This may be somewhat remarkable for some, because it is common practice to speak about ‘syntax and semantics’ as if they are two non-overlapping aspects of language. However, semantics is concerned with the relation between expressions and states of affairs, and therefore they include the syntactic structures of these expressions. Defining semantics, therefore, includes defining syntax, and sharing semantic conditions implies sharing definitions of syntactical structures.

Now some elaboration on semantic equivalences follows. Semantic equivalences expressed in a formal-language define which expressions are equivalent in interpretation to other expressions, in such a way that this information can be fully used by automated-deductive-reasoners. These equivalences require the inclusion of formal-language constructs to express them. These are formal-language-constructs, which, in a sense, have a reflective nature. They state something about the semantics of other formal-language-expressions.

Example 6. *Example: $c_0 == c_1$ expresses that constants c_0 and c_1 both refer to the same entity. Combined with example 4 (p. 184), this will enable the given inference.* ■

III.3.5.6 Formal-languages as sets of semantic conditions

The aforementioned insight that a formal-language is equivalent to a set of semantic conditions, including its domain-related-semantic-conditions, has some implications for the boundary between a language and its usage. It implies that RDF+S, first-order logic, OWL etc. are, when they are used to represent world-knowledge, in fact not complete languages, but sets of structural-semantic-conditions. The semantic conditions are extended by its users at the moment that they express new information,

for example by introducing new nodes in RDF+S. In such a case, the users are actually extending the formal-language, and not merely using it. Contrast this with the assumptions that underly statements made by some that a user is merely *using* OWL to represent a given state of affairs. The assumption seems to be that the user is not, or does not have to extend the language to be able to do so.

I believe that the insight is not only philosophically elegant, but may also have practical consequences for reducing fragmentation. A natural language is associated with a complete set of semantic conditions, and even more. It is complete in the sense that both the structural and semantic conditions are such that any valid sentence is grounded and expresses a states of affairs in the real world. *Example: Compare the sentences $\forall x : (M(x) \rightarrow P(x))$ and “all men wear a purple hat”. The first merely describes a structure, the second describes that same structure, but also establishes a connection with an actual proposed state of affairs in the world.* In the scope of formal-language the sets of conditions are mostly of an incomplete nature. They are often indeed merely describing structural-semantic-conditions. If people confuse these different meanings of the word ‘language’, they may be inclined to start new formal-languages from scratch, instead of building upon the good work that already has been done. An expert in KR-languages will be aware of this difference, however, the approach in this book towards fostering formal-fluency is to lay the foundation of approaches that also attract non-specialists to help in developing the composition-record including the formal-languages and their semantic conditions.

Another implication of the necessity of domain-related-semantic-conditions to represent real-world knowledge is that full formalisation does not exist. After all, these conditions establish a relation between language constructs and notions within a given domain that exist beyond the domain-related-semantic-conditions. This also holds for any language that has the appearance of being strictly formal such as one meeting the conditions of first-order logic. When first-order logic is *applied* to represent real world knowledge the meaning of predicates (e.g. ‘being in Paris’) and constants (‘the Eiffeltower’) is grounded in the realm of human experience, and cannot be expressed solely by using the semantic conditions of first-order logic. A person reading the representations may therefore feel that the expressions imply more than is captured by the structural-semantic-conditions.

III.3.5.7 Efficiency of formal-language representations

III.3.5.7.1 Overview

Typically, a given piece of information can be expressed in different ways in a given formal-language. The efficiency with which a given reasoner can operate may be different depending on the chosen way. This implies that a challenge is how to foster

the capability to choose representations for which the applied reasoners perform well.

III.3.5.7.2 Relevant assumptions

The relevant assumptions to focus on the given challenge are as follows.

EfficiencyGivenReasoner = Ideal

$$+ \{ \text{-ideal-reasoner-oracle} + \text{unique-reasoner-per-formal-language} \} \quad (\text{III.3.5})$$

III.3.5.7.3 Relevant properties

A relevant property of this focus is best illustrated by an example.

Example 7. *Suppose that the ideal translator oracle translates a given state of affairs into FOL, by adding to a KR-base Γ the sentence φ with the following structure: $\varphi_0 \vee \varphi_1 \rightarrow \varphi_2$. Here, φ_0 , φ_1 and φ_2 are sentences as well. Let reas be the unique reasoner that is available for FOL. Now, suppose that both φ_0 and φ_1 are deducible from Γ . Also, suppose it takes reas a billion years to deduce φ_0 , while it takes only 1 microsecond to deduce φ_1 . Suppose that one applies reas to establish whether φ holds or not (the query or reasoning goal). If reas evaluates from left to right and ‘eagerly’ (depth first), then reas will need a billion years to solve the problem. If the same state of affairs would have been translated into the equivalent sentence $\varphi_1 \vee \varphi_0 \rightarrow \varphi_2$ instead, then reas would have finished in 1 microsecond.*

■

This example demonstrates that the way a state of affairs is expressed in a formal language, can have a tremendous influence on the efficiency with which a given reasoner can be applied. (In spite of the fact that these expressions are semantically equivalent.) This holds even for minor differences.

III.3.5.7.4 Potential development decisions

First, the more people know about the peculiarities of the unique reasoner of each formal language, the better. Second, tools could be created that carry out optimisations of the expressions automatically. These tools enable one to apply each unique reasoner more effectively, and thus, the score on the metric-for-integration-of-automated-deduction will go up. (The question, however, is whether this is completely legitimate given the assumption set. In a sense these tools create modifications

of the existing reasoners, because they can be regarded as a kind of ‘pre-automated-deductive-reasoner’. The reasoners, therefore, would cease to be unique for their formal-language.)

III.3.5.8 Selection of automated-deductive-reasoners

III.3.5.8.1 Relevant-assumptions

In a scheme, the assumption-set is as follows:

ReasonerSelection = Ideal

$$+ \{-\text{ideal-reasoner-oracle} + \text{fixed-reasoners-per-formal-language}\} \\ + \{-\text{formal-super-language} + \text{one-formal-language}\} \quad (\text{III.3.6})$$

III.3.5.8.2 Relevant properties

Because of fixed-reasoners-per-formal-language: all automated-deductive-reasoners may perform differently depending on the reasoning-goal and the content of the KR-bases. Therefore, the choice of automated-deductive-reasoner has an influence on the performance.

III.3.5.8.3 Metrics and requirements

Consider a function that maps the content of a KR-base to an automated-deductive-reasoner choice, the *reasoner-chooser-function*. The more this function approaches the reasoner-chooser-oracle, the higher the quality of the selection process. This requires good knowledge of the existing automated-deductive-reasoners. *reasoner-chooser-function*

It is indeed so that the score on the metric-for-integration-of-automated-deduction will increase when this sub-quality metric increases in value.

III.3.5.9 Further areas of focus

As said before, it is not the purpose of chapter to create a final survey of all challenges. Next to the challenges that have been worked out so far, the following subsections cover, in less detail, a selection of other major challenges in the fostering the integration-of-automated-deduction.

III.3.5.9.1 Dealing with changes in information

The information contained in a community is of course continuously evolving. Consequently, the information expressed in KR-bases evolves as well: new expressions

are added, old ones are retracted. If we assume consistent information to hold, a major development challenge here is how to incorporate these changes so that automated deductive reasoners take them into account as quickly as possible. One of the subchallenges is that there are automated deductive reasoners that optimise their search algorithm in a time-consuming process by analysing the content of the given KR-bases. After the optimisation, it is not possible to add new facts without going through that time-consuming process again. So, here there is a trade-off between how quickly an automated deductive reasoner can keep new facts into account, and the reasoning speed per question.

III.3.5.10 Balancing the trade-off between quality and speed

The ideal of a maximised automatically deducible fraction (section III.3.5.2 (p. 174)) can never be reached for the collective body of information expressed in the world: it can only be approached. This also holds to a lesser degree for smaller bodies of information. An important reason is that there is a trade-off between the speed at which a person can create a formal representation and its quality. Among other things, this will be visible in the degree of granularity of the representation. For example, when using EERGs (defined in section III.5.4.1) or RDF+S, the same chunk of information can be represented with a few nodes, each carrying a complex meaning, or with many nodes with a more elementary meaning. The first will often be quicker to define, the latter will approach the ideal better.

III.3.5.11 Evolution of knowledge models

The collective body of information created by people consists of information from different communities for different purposes. These communities do not use perfectly overlapping knowledge models and semantic conditions. Even with the best of effort, it is reasonable to assume that it takes time for these models and conditions to evolve towards each other. This process never ends, because many, if not all, domains are in a continuous process of change.

III.3.5.12 Dealing with rapid changes in information

Some information expressed in KR-bases changes so quickly and massively, that it is difficult for automated deductive reasoners and data transport mechanisms to keep up. An example is weather information, which may change from second to second and have a tremendous volume. The same holds for much other sensor data, such as human-transportation traffic data, data from experiments, etc. Potential development

decisions here could make use of distributed reasoning algorithms, to relieve the data transport mechanisms.

III.3.5.13 Development of new automated deduction techniques

Given the analysis that there is no best automated-deductive-reasoner and formal-language, it would be advantageous if the set of available reasoners and formal-languages could be extended (indefinitely). This is needed if something occurs that cannot be represented or reasoned about effectively, or not at all in the available formal-languages and reasoners.

This requires the collective capability to develop new techniques, and not only make use of the existing repertoire of instruments to represent and automatically reason about information. A problem is that the addition of new automated-deductive-reasoners and formal-languages can also contribute to more fragmentation. Counter-measures have to be taken to minimise this.

Potential development decisions are (1) Educate more people in being able to develop automated-deductive-reasoners and formal-languages. (2) Develop better educational material to achieve this. (3) Improve and create tools so that less expertise is required to develop automated-deductive-reasoners and formal-languages, or so that these can be developed more speedily. (4) Put instruments in place which prevent formal-language fragmentation as much as possible if new automated-deductive-reasoners/language are added, see section III.3.5.5 (p. 184).

III.3.5.14 Trust

Establishing trust, here in the sense of the reliability of information expressed in KR-bases, is an inescapable challenge in larger heterogeneous environments. When can a user trust the information expressed in the KR-base by other people or their machines? These other actors may make mistakes, or deliberately express information faultily. Potential development decisions are the development and application of trust metrics that are based on the reputation of the actor who added the given representation.

III.3.5.15 Inconsistencies

Inconsistent KR-bases are an inescapable problem in larger heterogeneous environments. Inconsistency may be caused by a difference of opinion, but also by mistakes. The development challenge here is to allow automated-deductive-reasoners to reason 'reasonably' in spite of the existence of these inconsistencies. In classical logic this is impossible because of the 'principle of explosion' (Carnielli and Marcos, 2001): if there is an inconsistency in the KR-base, everything becomes deducible, and as such

rendering the results of the automated-deductive-reasoner devoid of information. Potential options are selecting consistent subsets based on some trust metrics (so here it interacts with the development challenge around trust. See section III.3.5.14 on the preceding page).

III.3.5.16 Privacy and security

If formal-language representations express sensitive information, the degree to which the system can provide privacy and security becomes an issue. The current overall class of metric, *metrics-for-integration-of-automated-deduction*, does not yet capture this challenge. This may imply that it is still insufficient. It should also integrate a metric that captures the degree to which one can secure one's information expressed in KR-bases. It may be obvious that such a metric is in a trade-off relation with *metrics-for-integration-of-automated-deduction*.

III.3.5.17 Future work

Future work may include: (1) As said, this survey about the challenges surrounding the integration-of-automated-deduction into society is intended as a continuous work in progress (section III.3.3 (p. 162)). The involved community of researchers, developers and other interested actors are invited to further fine-tune metrics proposed in this chapter, and apply them as valuable instruments to guide the direction of future development efforts. (2) Application of the *layered-system-approach*: Present the survey divided into layers where the top-level layer contains the most abstract and general description of the system, and each subsequent layer is a more detailed specification or implementation of the preceding layer. This is the *layered-system-approach* formerly introduced in section II.4.2 (p. 52). It is intended to contribute to a more intelligible survey, among other things, by revealing manageable sets of subchallenges layer-wise instead of all at once.

Chapter III.4

Coba extensions for fostering formal-fluency

III.4.1 Introduction

This chapter investigates the compatibility of the Orcoba-Approach with fostering formal-fluency and then briefly introduces the specialised extensions that I have initiated for this purpose.

III.4.2 Compatibility

This section briefly focuses on the compatibility of the Orcoba-Approach and some of its general extensions with fostering formal-fluency. With the latter I mean to investigate whether it is plausible there indeed exist a possibly unlimited number of effective extensions, without providing particular examples as of yet.

First, one of the prerequisites for the application of the Orcoba-Approach to a capability is that it should be possible to express the method to practice this capability in a permanent form, so that it can be captured in a composition-record. This is, I believe, the case for formal-fluency. Formal-fluency is based on formal-languages, which have been developed with the exact purpose to objectify and formalise the expression of knowledge. Therefore, formal-fluency lends itself well for expressing the method to practice it in a written form. Experimentation with some extensions indeed showed that participants were able to use the written composition-records quite naturally, and were able to solve the challenge purely by using these written

composition-records.

The Challenge-Based-Orcoba, and in particular the Or-evohut are extensions of the Orcoba-Approach that have been designed to make fostering human capabilities much more efficient. However, applying these to a certain capability, requires the definition of a suitable challenge-set, and a corresponding challenge-assessment (see section II.4.5 (p. 57) of part II (p. 21)), which may not be easy or feasible for all types of (collective) capabilities. Formal-fluency is intended to produce very precise representations of information that, by definition, can be sensibly manipulated by computer programs (automated-deductive-reasoners). Exactly because of this fact, the existence of suitable challenge-sets seems to be very plausible. I believe the challenge-sets defined in the subsequent chapters support this.

III.4.3 Extensions

The specialised extensions of the Orcoba-Approach for the purpose of fostering formal-fluency developed in this work are System- α -for-real-time-collective-formal-thinking and SWiFT (Semantic Web in Fast Translation).

In real-time-collective-formal-thinking a community collaboratively solves a problem, while all members express their thoughts about the problem at the moment they conceive of them in a fine-grained way in a persistent shared digital collective memory (a KR-base) using a formal-language, according to a predefined set of rules. (A KR-base is a database with knowledge representations in a formal-language.) System- α -for-real-time-collective-formal-thinking is the first concrete Orcoba-system to foster real-time-collective-formal-thinking developed in this study and used during experimentation. Orcobas for fostering real-time-collective-formal-thinking are the topic of chapter III.5 (p. 197).

The SWiFT-class-of-games incorporates a competitive multi-player online game element into the Orcoba-Approach, among other things to motivate participants and maximise accessibility of SWiFT to many, accelerating the evolution process. SWiFT proceeds, in brief, as follows: Competing teams translate the same natural language text into a formal representation, each player covering a fragment that does not overlap with others of his team. Then teams challenge each other's translations by posing questions that have a very specific answer based on the content of the text (question attack). A question could for example be: give me all vitamins that act as a catalyst for cell growth. Each team then tries to construct a formal query that deduces the answer from their translation (algorithmic defence). A short translation time, a low average complexity of queries, and correct answers contribute positively to the final score of a team. Chapter III.6 (p. 215) elaborates further on the SWiFT-class-of-games.

The following extensions and variants of SWiFT are explored in this part: SWiFT-Full (chapter III.6 (p. 215)), SWiFT-Fixtal (chapter III.7 (p. 225)) and higher-order-SWiFT (which includes SWiFT-Focused) (chapter III.8 (p. 245)).

Chapter III.5

Real-time·collective·formal·- thinking

(Before starting this chapter, I would like to gently remind the reader of the call for feedback on page i.)

III.5.1 Introduction

If one seeks to increase integration-of-automated-deduction in a community it would be ideal if *all* relevant information that appears within a community were expressed optimally in a deductive-formal-language – by means of some magical ‘oracle’ (also see section III.3.5.1 (p. 172)). To the best of my knowledge, unfortunately, such an oracle does not exist.

Rather, one has to come up with inventive ways to at the least get somewhat closer to this ideal. A central question in this quest is: where and when exactly is new information created by people? The answer, obviously, is: in the minds of people, in the form of thoughts. If we would be able to foster people’s capability and motivation to express many of their relevant thoughts when they first conceive of them immediately into a formal-language, then a large portion of valuable information produced is transformed right at its source. This may be an effective way to formalise a large portion of the collective body of knowledge produced by a community – and keep these formalisations up-to-date. An analogy is the following. Suppose the protagonist of a movie tries to save the entire population from certain death caused by an alien virus. Fortunately, there is a remedy: a water soluble medicine. The heroine, clever as she is, releases this remedy at the source of the water people drink

from, of course after epically defeating aliens guarding this source. This way she ensures that the remedy will reach almost everyone. This will be far more effective than trying to get the medicine into the water after it already has been distributed. In a similar way, it would be quite effective to transform information at the places it is created, one of the major places being in the thoughts of people.

*real-time--
collective--
formal-thinking*

It would be nice to create a ‘micro-setting’ in which this process of ‘thought transformation’ can take place in an accelerated and readily comprehensible fashion. This allows effective experimentation with different approaches to foster the process. This study has conceived such a micro-setting, for which I coin the term *real-time-collective-formal-thinking*. In *real-time-collective-formal-thinking* a community of people collaboratively solve a problem, while they express their thoughts about the problem in a fine-grained way in a persistent shared digital collective memory (a KR-base) using a formal-language, at the moment they conceive of these thoughts, according to a predefined set of rules. One of the central rules is that the formal-language constructed and used by the participants is *shared* to the greatest extent practicable.¹ Their means of communication about the topic is essentially limited to the information expressed in the formal-language. Note that an essential reason to choose a community, rather than an individual, is that it necessitates the creation of a shared formal-language. *Real-time-collective-formal-thinking* is a specialisation of the broader *collective-formal-thinking*, in which real-timeness of translation is not required.

*collective--
formal-thinking*

*Semantic
Network*

The specific setting for *real-time-collective-formal-thinking* described in this chapter is *System-Real- α* (*System- α -for-real-time-collective-formal-thinking*). *System-Real- α* consists of a small team of people (composed of 2 to about 15 individuals) to contribute to the aforementioned comprehensibility and manageability of the experiment. *System-Real- α* ’s formal-language is based on expressing a web of relations between entities, a so-called *Semantic Network* (Sowa, 1992). *Semantic Networks*, however, can be constructed in many ways. Only a small number of them are highly reusable in other contexts, than the context in which the information was expressed, both for direct human usage, and for automated-deductive-reasoners.

As stated similarly before in section III.2.6.1 (p. 145), a note to put things into perspective: I neither expect nor pursue large scale adoption of *real-time-collective-formal-thinking*, at least not in the near future. Rather, systems for *real-time-collective-formal-thinking*, are, among other things, intended as an ‘incubation chamber’ for enabling a group of forerunners to lead the way for others to follow, and adopt aspects of the fruits of the results. In fact, certain design ingredients are not even intended to become parts of some ultimately adopted approach. They are ‘artificial’

¹Section III.3.5.5 (p. 184) describes this ‘sharing’ rule in more detail. It uses the more precise term ‘minimisation of formal-language fragmentation’, and shows that this contributes to the more essential goal of optimising the so-called deductive-closure.

ingredients, intended to improve the acceleration effect. For example, in spite of the fact that during the experiment with System-Real- α all communication goes through computers, I do not envision communities restricting their communication to this channel only. I consider other means, such as face-to-face meetings, valuable and irreplaceable.

From now on, the text will employ notions from the Orcoba-Approach as they have been developed in chapter II.3 (p. 43). In terms of the Orcoba-Approach:

1. System-Real- α 's pursued-capability is real-time-collective-formal-thinking in a small team setting.
2. System-Real- α has an initial composition-record, which describes, among other things, the rules participants have to follow, and
3. it also comprises an Orcoba-extension that among other things, consists of Constitution-Based-Subleme which will be briefly introduced later.

This chapter further defines System-Real- α . Experimentation with the system can be found in chapter IV.2 (p. 315).

III.5.2 Related work

To position this work among related work I want to make use of the following dimensions: the degree to which the expressions (1) are created by the creators of the knowledge, (2) are optimal for automated-deductive-reasoners, (3) are understandable for other people than the creator(s) (in their unaltered form), (4) are created in real-time (so, for example, at the moment the information is first conceived), (5) help their creators grow in their understanding of a topic, (6) require a minimal amount of training for the average person to be able to write and understand them. Ideally, the expressions produced score perfectly on all dimensions. However, as a consequence of trade-offs, specific choices have to be made. By means of these dimensions, I will now establish an important niche, which, to the best of my knowledge is not yet fully occupied by other strongly related work. The dominant focus of the Semantic Web research community (Berners-Lee, 1999) as a whole is optimising dimension '(2)', also called 'semantic interoperability of computer programs'. The focus is more on the 'under the hood' representation of knowledge used by applications, rather than on stimulating people to express their information directly in such a form. Therefore, it is considered more to be the task of the specialist (for example a programmer) to create the representations. Most people are mainly indirectly exposed to these representations, by an improved performance of the applications

they use, so dimension ‘(1)’ is not part of the Semantic Web community’s focus, while it is in this study.

In the area of Computer-Supported Collaborative Work there is a strong relation with work of Buckingham: Compendium ² and related systems such as Claimaker (Buckingham, 2007), and the complementary human capabilities implied by those tools. As far as I could establish their focus is on optimising ‘(1)’, ‘(3)’, ‘(4)’ and, perhaps more than the others, ‘(5)’. However, dimension ‘(2)’ is less clearly mentioned, and the emphasis is on use of predefined sets of relations and entities to be used by the participants (such as the ontology of Cognitive Coherence Relations), diminishes ‘(1)’. In my focus the participants create their own relations and entities. Buckingham has sacrificed ‘(6)’ to the benefit of the other dimensions, just as I do. I deduce this from the extensive training sessions that are organised by the Compendium team.

Another approach, implied by CMapTools³ (Cañas et al., 2004; Selevičienė and Burkšaitienė, 2016), seems to focus on optimising dimensions ‘(1)’ and ‘(5)’, but completely neglects ‘(2)’. The rules established for attributing meaning to the expressions are so vague, that ‘(2)’ is minimal. For the same reason, I doubt whether ‘(3)’ is really optimised in this approach, in spite of the fact that one has developed software to share representations.

In the area of Semantic Wikis (Bry et al., 2012), a confluence of the Semantic Web and Wiki paradigms, there seems to be a predominant focus on optimising ‘(1)’, ‘(2)’, ‘(4)’ and ‘(6)’. An element that works against dimension ‘(2)’ in Semantic Wikis, is that conventional representation in natural language documents still dominates. The alternative way of expressing information is regarded more as an enrichment (‘semantic annotations’) than as a first-class citizen as it is in my approach. I expect that the human capability that is subject of this chapter would sooner reach a high degree of maturation under my conditions.

System-Real- α pursues optimising dimensions ‘(1)’, ‘(2)’, ‘(3)’ and ‘(4)’ as much as possible. For this purpose, ‘(6)’ is sacrificed to a great extent to the benefit of these dimensions. Again, I would like to stress that this does not imply I do not consider a maximisation of all as the ideal case. However, by sacrificing ‘(6)’ I hope to foster a community of knowledge workers that have an unprecedented capacity of maximising the mentioned dimensions. As said earlier, although not in my focus, I anticipate that they could form a community of forerunners that smoothen the path for others, so that ‘(6)’ may – in the long-term – be met to a much greater extent. The remaining dimension, ‘(5)’, is to a much lesser degree in my focus. However, dimension ‘(5)’ I believe to be the least in a trade-off relation with the others, but, in contrary, benefits from them in many cases. Therefore, I think that insights from

²<http://www.CompendiumInstitute.org>

³<http://cmap.ihmc.us>

work focusing on ‘(5)’ could in a later stage be integrated fairly easily in my approach. I admit, however, that I also believe there are several modes of expressing information that do not have an obvious computational sense, but that are nevertheless important for human understanding (such as visual sketches). These modes are beyond the scope of this chapter.

III.5.2.1 Overview

This chapter does not fully describe the initial composition-record (method of collaboration), but explains it only as far as necessary for the purpose of this chapter. Section III.5.3.1 provides a summary of the method and section III.5.4 on the following page covers the relevant aspects of the formal-language. In section III.5.5 (p. 206) follows an elaboration on the required properties of entity-definitions. In section IV.2.3 (p. 316). Section III.5.6 (p. 208) concludes with suggestions to improve the method and refine the focus of the study. Note that the associated experiment is covered in chapter IV.2 (p. 315).

The Orcoba-extension for System-Real- α includes the following aspects. After the participants have chosen a problem, they log into the software environment, coined Constitution-Based-Subleme (CBS)⁴. They immediately start translating each thought they have concerning the problem into an addition to the KR-base provided by CBS in a formal-language. There is no further communication between the participants.

III.5.3 Initial composition-record

III.5.3.1 Collaboration rules

During a session, the participants translate each thought they have concerning the problem into an addition to the shared KR-base provided by CBS. There is no further communication between the participants. Before a participant can express a relation between two entities (a triple), he or she has to create the three nodes with corresponding entity-definitions (the relation is also considered to be an entity) to build the triple, if they have not been created yet. Each of the entity-definitions must first be judged and accepted by all other participants before it can be used to construct the triple. After the triple has been created, all participants give their opinion about the created triple as soon as possible.

⁴Open source groupware environment developed by Chide Groenouwe, based on the work of Michiel Visser.

III.5.4 The formal·language

This section defines and motivates the formal·language used in the shared KR-base.

III.5.4.1 Exact·Entity·Relation·Graphs (EERGs)

*Exact-Entity-
Relation-Graph
EERG*

The foundation of the formal·language are *Exact-Entity-Relation-Graphs (EERGs)*, which are to be considered as a *sub-category* of Semantic Networks. An *entity* in an EERG is a specific mental or physical entity. Examples of descriptions that define an entity are: ‘the class of scientists’, ‘my bicycle’, ‘the number 6’, ‘the set of natural numbers’. Note: ‘the concept elephant’ is allowed, but ‘an elephant’ not (which elephant?); ‘the expression ‘number *N*’ ’ is allowed, but ‘number *N*’ (which number *N*?) not.

digital-entity

A *digital-entity* is an entity that is a digital piece of information. Examples: strings of symbols and mp3-files. Digital entities in an EERG have as semantics themselves. For example, the string “10” is not a number, but a row of two symbols “1” and “0”. Other concrete examples: string “10” and “ten” are two different digital entities, in spite of the fact there are perspectives in which they have the same interpretation. “10” interpreted as the (decimal) representation of the number ten, and “10” interpreted as the (binary) representation of the number are one and the same digital-entity. The string “To be or not to be” is not identified with the first line of Hamlet 3/1 of Shakespeare. It is identified with being a specific sequence of symbols. A *non-digital entity* is an entity which is not digital, for example the person Gandhi.

In an EERG, non-digital entities may be represented only by a node with a neutral identifier, such as a number, but not a meaningful name. (If one wants to add a description or a name to a non-digital entity, one can do so by attaching a digital-entity to the node with a suitable relation.) A node in an EERG that represents a digital-entity does not have any identifier, because it contains the digital information that makes up the digital-entity and therefore can be identified uniquely by its content.

Finally, an Exact-Entity-Relation-Graph is a directed graph in which each node represents an entity as defined previously, and each arc defines a specific relation between a pair of entities, including between an entity and itself. Relations are also entities, thus relations between them and other entities can also be expressed. The arcs are directed, because both nodes can play another role in the relation.

The most important motive behind using EERGs is that I believe it leads to a greater transparency of the information expressed than using Semantic Networks without these additional refinements. For example, in many other Semantic Networks (such as the Semantic Web) there is a confusion between the node that represents the entity and the labelname used for that entity. A related confusion

The text at the right defines which entity is represented by node with identifier 314. Note: for brevity, I will from now on use *{hasHumDescr}* as an abbreviation of *{hasHumanDescription}*.

Every description should start with the text {"■ represents the ..."}. The black square refers to the node of which the meaning is being defined, in this case the node with identifier 314. I have introduced this construction to reduce the probability that the participant does not define a specific entity. The black square (■) stresses that it is about a (mental or physical) entity, and the word "the" stresses that this thing must be specific. For example, it prevents "Elephant" being used as a description.

Important remark: since this format was not a requirement during the first three experiments, not all participants will have used it. I, however, will use it in my examples. Sometimes, I will omit the entity that is defined in order to save space, thus I will only write:

{■ represents the set of natural numbers.}

III.5.4.1.2.1 Preferred labels Preferred labels attached to entities are purely intended as a mnemonic aid for the participants for easy reference to the entities they created. Their exact requirements are mentioned in section III.5.5 (p. 206). The notation for preferred labels used in this chapter is provided in section III.5.4.1.1 on the preceding page.

III.5.4.1.3 Extensional meaning

Each description is to be interpreted by its extension only. The extension (also denotation) of a description consists of the objects to which it applies, in contrast with its comprehension or intension, which consists very roughly of the ideas, properties, or corresponding signs that are implied or suggested by the description (Carnap, 1956). For example: the description {"■ represents the set of sons."} defines exactly the same entity as the description {"■ represents the set of men."}, for each entity is a set that contains exactly the same objects. For a description that refers to physical entities, the extension is intuitively quite easy to grasp. Examples are persons, bicycles and African masks. The meaning is simply the real objects to which the description refers, nothing more and nothing less. For descriptions that refer to mental objects, such as numbers and scientific theories, it is much harder to define the exact meaning of "extension". This question can be brought back to which mental entities are allowed to exist, and which are not. Further research has to be done on this topic.

III.5.4.1.4 Desired and pursued properties

To give another important motive behind the core properties, I will first introduce the distinction between desired and pursued properties. To approach real-time translation of thoughts into EERGs, it is crucial to establish entity-definition properties that can be pursued in real-time by at least a significant body of trained participants. On the other hand, ideally each entity-definition should exhibit the highest degree of reusability. These two demands are obviously in a trade-off relation.

I call properties pursued by a participant during the creation of entities-definitions pursued properties. Properties the composition-record-developer desires to hold true for the resulting entity-definition I call the desired properties. Although it might be so that desired properties and real-time pursuable properties do not coincide, it might be possible to choose the latter in such a way that they indirectly lead to or at least approach the desired properties. The following section defines one of the most fundamental desired properties of entities.

III.5.4.1.5 Desired property: minimisation of elementary-nodes

In the method there are two ways to attach meaning to EERG nodes. First, by defining them in terms of natural language. Essentially, one is defining a term from one language (a node is in fact a term in the EERG language) as a combination of terms of another language (natural language) of which the meaning is already known by the participants. I call these nodes *elementary-nodes*. Second, by defining them in terms of other nodes in the graphs, which already have a meaning attached to them. I call these nodes *composite-nodes*.

*elementary-
node
composite-node*

Previous concepts allow the formulation of one of the most important desired properties of entity-definitions: The elementary-nodes should be as reusable as possible. (Note that this is not the only desired property. There are more desired properties defined in the complete description of the method.) In other words: the elementary-nodes to express the information up to a certain moment should be chosen in such a way, that for any future expression of other information from any context, the total number of new elementary-nodes needed to be added remains lowest as possible. *Example: {“■ represents the class of loafs of bread.”}. Non-example: {“■ represents the class of loafs of bread baked at a temperature of 90° C in olive oil.”}. Some important motivations include: (1) This increases one type of reuse: many more meanings can be constructed with entities with this property. (2) When using a small base of entities, the probability is higher that the same or related information will be expressed using the same entities, increasing the likelihood that people will find information related to their topic more quickly, preventing them from doing the same work more than one time. (3) I assume that the manipulation by computer programs*

is also easier, because more of the semantics will be reflected in the structure of the graph.

III.5.5 Core properties of entity-definitions

The four required core properties of entity-definitions in the initial composition-record are given in the following paragraphs.

Article 1 (<Unambiguous>). *The entity to which the node refers is defined as unambiguously as possible, in the first place for the other participants and in the second place for the widest audience possible. For example, the description should not presume knowledge about other entities or relations in the graph to be understood. Motivations: (1) it increases possibilities of reuse, because it allows reuse of the same node, even in other contexts, without the necessity of changing it; (2) it is easier for others and the participant's future self to understand which entity it's about.* ■

Article 2 (<ExtensionalMeaning>). *The participants should interpret each description by its extension only (see section III.5.4.1.3 (p. 204)). In other words: when all descriptions in the graph would be replaced by whatever description with the same extension, no information may be lost.* ■

Article 3 (<GenericDescription>). *The description of the entity should be as generic as possible, but not so that its extension denotes another, broader, entity than intended. For example: {"■ represents the category of sons."} should be replaced with {"■ represents the category of men."}, (because each son is a man and vice versa) however, {"■ represents the category of living beings."} would be too broad. Another example is:*

- {117}—{hasHumDescr}→{"■ represents the number of inhabitants of Japan at January 1st 2007."} (Warning: {117} does not represent the integer 117, but the entity to which the identifier 117 is assigned within the EERG.)

The most obvious extension of the description given above is a specific integer. Therefore, the given description is not a generic one. In contrast, the number described by a decimal notation would have been generic. Suppose the number of Japanese people at that moment was 125400113. In agreement with the method would have been:

- {117}—{hasHumDescr}→{"■ represents the integer with decimal representation 125400113."} (Warning: {117} does not represent the integer 117, but the entity to which the identifier 117 is assigned within the EERG.)

However, what should one write down when one does not know that number? Moreover, even if one would know that number, the information one intended to express, namely

information about the Japanese population, is lost. The following EERG solves this problem in a way that is in agreement with the method:

- {134}—{hasHumDescr}→{"■ represents the set of inhabitants of Japan at the 1st of January 2007."}
- {147}—{hasHumDescr}→{"■ represents the relation that is defined as follows:
 x —{hasHumDescr}→ y if and only if x is a set (as defined in mathematics) and y the number of elements in that set."}
- {134}—{147}→{117}

Note: this is not a single node. There is no single node possible that expresses the intended information. This provides a clear example of the distinction between pursued and desired properties: pursuing the property <GenericDescription> contributes positively to the desired reusability of nodes (see section III.5.4.1.5 (p. 205)).

Motivation: with this property, a definition is more likely to be recognised from other contexts, which increases the reuse of the node. ■

Article 4 (<SingleEntity>). The description introduced defines a single, specific, entity. It is not a parameterised description for defining many entities. Thus, the description {"■ represents the integer n ."} does not describe a single entity, but is a parameterised description for a collection of entities (to wit: {"■ represents the number 1."}, {"■ represents the number 2."}, {"■ represents the number 3."}, etc.). Instead, depending on the context, one could have defined {"■ represents the set of integers.."} or {"■ represents the function with definition: $f(n) = n$."}, which are both descriptions of specific entities. Motivations include: parameterised descriptions often imply context, while entities should be defined in such a way they can stand on their own to greatest extent possible, which in turn optimises possibilities for reuse in other contexts. E.g. {"■ represents the real number x^2 ."} does not have any meaning without a context in which is spoken about a specific number, while {"■ represents the mathematical function on real numbers with prescription $f(x) = x^2$."} does have a meaning as a separate entity. Note, however, that a parameterised text regarded as an expression is allowed. E.g. {"■ represents the expression 'the real number x^2 :'} this refers to the single entity that is an expression. ■

Article 5 (<LabelRequirements>). One is free to define any preferred label, as long as the following conditions hold: (1) when all labels are erased from the EERG, no information may be lost (other than the fact that the community used that specific set of labels as a mnemonic aid). (2) it must be unique throughout the graph, and one should keep into account that it remains unique also when future entities would be defined. A non-example: {"777"} as preferred label of the mathematical triple (7, 7, 7) (confusable with the integer notation 777, which might be used in future).

Motivations include: without the first point, one can use labels to ‘smuggle in’ information that should have been expressed with the definition of adequate high quality entities and relations. ■

III.5.6 Discussion

III.5.6.1 Improving focus

A major limitation of System-Real- α for the purposes of fostering formal-fluency is that the selection of specific difficulties in writing EERGs that are encountered depend on the specific route the participants follow during their collaboration. I became fully aware of this during experimentation (see section IV.2.4 (p. 320)). They might, for example, never encounter a thought to be expressed for which the temptation to define a parameterised description is high. This problem could be solved by temporarily limiting the freedom of participants by a series of small experiments: for example by choosing a set of natural language texts to be translated that confronts the participants with certain pitfalls, instead of letting the participants translate their own thoughts.

Related to this is the following more general observation. Formal-fluency, the central topic of this part of the work, is an essential ingredient of real-time-collective-formal-thinking. However, real-time-collective-formal-thinking is not equivalent to it, but a broader, more encompassing capability than formal-fluency. Therefore, it demands more of the participants than formal-fluency. This has the disadvantage that it does not isolate that specific capability, but the advantage that it fosters the community’s capability to apply formal-fluency in a highly important context. The relation between real-time-collective-formal-thinking and formal-fluency is perhaps somewhat analogous to the relation between the ‘biceps curl’ and the ‘pull up’ physical strength exercises. A ‘bicep curl’ is an isolated, and highly effective exercise for the biceps, while the ‘pull up’ is an exercise in which the biceps are used in a more complex coordinated movement between several muscles. The first may be most effective for training your biceps. The second, however, may, depending on the context, be much more useful for contributing to the overall functioning of your body. In practice, probably an athlete needs to mix both types of exercises, and find a balance between isolated training and several forms of in-context training.

The scope of this work, however, is to foster formal-fluency. The subsequent sections, therefore, develop variants of a special game (SWiFT). These games isolate formal-fluency much better, similar to what the biceps curl exercise does for biceps.

Another, related problem arises from the fact people are allowed to express the content of thoughts as separate natural text entities in an EERG. Allowing this creates

another shortcut around the core of the composition-record: expressing thoughts in a reusable way. The same can be said for any ‘information entity’ (such as documents and audio files). These can express information that should have been expressed as a combination of elementary-nodes and relations.

However, it is important to allow this shortcut so that participants can decide for themselves whether they want to invest energy in creating a fine-grained representation, or create a less fine-grained representation to be refined incrementally. SWiFT, developed among other things in response to this experiment, adequately deals with this problem, by defining a metric that is directly based on the degree to which automated-deductive-reasoners can be applied to infer answers from the formal-language expressions created by the participants from a fixed piece of natural language text. This makes it attractive for participants to balance the mentioned trade-off wisely.

III.5.6.2 Changes to the contributive-aspects

During experimentation and further contemplation, ideas for improvement of the composition-record arose. Each of the following points suggests improvements to the composition-record to reduce the probability that human fallacies will occur. Hence these fall in the category of the contributive-aspects of the composition-record. See section IV.2.4.2 (p. 320) about the experiment for more details.

1. (a) The participant should pay extra attention to **<SingleEntity>** when a parameter occurs in a human description (such as for example x , N , q , etc.).
 - (b) The software scans for symbol sequences in the human description that could be parameters (this can be checked quite easily on a syntactical level) and warn the participant to double check **<SingleEntity>**.
2. The participants are made aware of the following pitfall. When solving a problem, one’s thoughts often unfold in sequence of (sub)questions and (sub)responses (or (sub)problems and (sub)solutions). A pitfall in this process is that one might define entities that form the ‘response’ to a question such that the entities that form the ‘question’ are needed to understand the entities that form the ‘response’. For example, given problem context “the number of orders in which N different objects can be placed”. With this in the back of one’s mind, one might easily be tempted to define the ‘entity’ {“■ represents the number $N!$.”}, which is not a single entity.”
3. (a) The software first lets the participant define the human description, and after that gives the option to attach a label, too. Currently, the order is

exactly the opposite, and on top of that, a label is obliged.

- (b) The software only allows an entity description to be submitted for judgement after the participant has run through a check-list based on the properties to be pursued.
- (c) Only after a description has been accepted by all participants, may a participant optionally create a label.

III.5.6.3 Changes to the constitutional aspects of the method

Further improvements to the constitutional aspects of the composition-record, so the essential aspects of it, may include:

1. Problems may not be defined as entities. In fact, an information-related problem can in most (if not all) cases be decomposed into two elements: incomplete information and an intention to complete that information. The second turns the first into a ‘problem’. Transparency would be increased when these two elements are separated explicitly. For example: instead of posing the problem “Who is the writer of Tao Te Ching?” (which is both a set of incomplete information as an intention indicated by the question form), one can reformulate it to intentionless incomplete information: “The writer of the Tao Te Ching is ...” (or in EERG form: $\{The\ Book\ Tao\ Te\ Ching\} - \{has\ been\ written\ by\} \rightarrow \{77\}$, in which {77} represents the writer of the book, unknown as yet). The participant can express the intention of it being a problem by indicating the pieces of information he would like to see completed, in the example by adding an extra pointer to entity {77}. This contributes to reusability at least in the fact that the formulation of answer and reply are one and the same.
2. Defining labels is optional. In some cases, a label is not necessary. Forcing participants to give a label anyway, could lead to several undesirable situations, including: labels being (almost) equal to the description, which is superfluous, or worse, labels containing additional information which should have been expressed with additional entities.

III.5.7 Future work

This section adds to the future work already suggested in section III.5.6 (p. 208). Moreover, this section broadens the scope from a focus on formal-fluency during real-time-collective-formal-thinking to all that is needed to facilitate real-time-collective-formal-thinking. However, it is beyond the scope of this section to provide an

exhaustive list of essential future work within this broader scope. Among other things, further experimentation with systems for real-time-collective-formal-thinking will be needed to approach the creation of such a list.

III.5.7.1 Formal-languages with higher expressivity

The formal-language in System-Real- α has a very limited expressivity. Formal-languages with a higher expressivity are desirable. Moreover, the translation produced by the participants has not been tested with respect to the effectiveness with which automated-deductive-reasoners can be applied to deduce valuable information. This problem is addressed in subsequent systems described in this book: SWiFT, among other places described in section III.6.4.1 (p. 218).

III.5.7.2 Version management

Information expressed in the shared KR-base can get outdated. This includes information related to the underlying ontology, i.e. the definition of the language to express states of affairs. For example, one's understanding related to the KR-base's content may evolve over time. As a consequence, one may desire to express the same information in a different way. Directions for a solution could include that former ways to organise expressions of information is never deleted, however, that successive ontologies are supplemented with meta-information that allows anyone to filter a suitable version. Challenges in this process include how to migrate information expressed in previous ontologies to a new version.

III.5.7.3 Further integration of the Orcoba-Approach

System-Real- α already contains aspects of the Orcoba-Approach. Among other things, it contains a composition-record. However, the most effective extensions of the Orcoba-Approach developed so far have not yet been integrated, nor did experiments take place in which participants updated the composition-record.

In particular, future work may include an integration of Or-evohut. However, this is not easy. For one, Or-evohut requires a well-defined scoring procedure. This also holds for the more general class of Challenge-Based-Orcobas. It is difficult to define such a procedure for real-time-collective-formal-thinking, because certain random variables related to real-time-collective-formal-thinking are hard to control. That is, there may be other variables than 'the quality of the real-time-collective-formal-thinking process' that influence the outcome of several scores. Consider the following scoring procedure: the time a team needs to solve a given problem while employing real-time-collective-formal-thinking. The problem with this score is that

speed can be attributed to many other factors than expressing and sharing one's thoughts in a formal-language and employing automated-deductive-reasoners. For example, a team-member may be brilliant and see the solution immediately so that hardly any formal-fluency has to be applied. I believe that it is not easy to control for 'brilliance', among other things, there are so many different types of it.

A solution, of course, is to compensate for these uncontrolled variables with large sample sizes. However, that is not particularly economical. It will become very costly to evaluate the quality of a given composition-record. A more promising direction would be to extend the scoring procedure, or to add conditions that should be met by the team. For example: look at the amount information that is expressed in the formal-language, and disqualify results that remain below a predefined amount. Additionally, ask participants whether they indeed used the expressed information to solve the problem, and did not just express it because it was an 'obligation' to do so. In particular one could look at whether information provided by the other participants was crucial for the next steps in solving the problem. Of course, it is not easy to assess this reliably, because it depends on the participant's testimonies.

Also, this score does not keep into account other benefits of real-time-collective-formal-thinking, such as a better reuse of information produced during collaborative problem solving for other, future problems to be solved by other people. This could also be included as a scoring procedure. For example: the score of a team is partially based on the success that future teams have with the KR-base they produced to solve similar problems. Note that therefore, this part of the score can only be assigned some time after the session ended.

III.5.7.4 Coordination of the real-time-collective-formal-thinking process

System-Real- α does not facilitate the coordination of the collective thinking process. This coordination constitutes the availability of instruments to express meta-information that can be used by human participants to establish where their thinking effort should go first. Such instruments could consist of standard ways to express such meta-information and directions, and defining a set of related protocols. For example, there could be a uniform way to express a question in an EERG that a participant thinks is crucial for solving the overall problem. Other participants can then filter questions from an EERG using reasoners, and use that as a way to establish where their thinking effort best should go.

III.5.7.5 Decentralised development of personalised educational nodes

I believe it is crucial that future systems for real-time·collective·formal·thinking facilitate participants to effectively learn the information expressed in a given EERG. The more people understand the content of an EERG, the more people can contribute to growing collective insight into matters. This may also lower the threshold for anyone to participate, even non-professionals in the domain that is covered by the given EERG. This could, for example, become a major contribution to citizen science. Citizen science is a “form of collaboration involving active engagement of members of the public in scientific projects which address real world problems” (Wiggins and Crowston, 2011; Morzy, 2015).

The problem is that merely representing information in an EERG will often not be enough for many participants to understand what has been expressed. This will even be the case if all the information is expressed in great detail and precision. In other words, nodes added by participant p_0 may be conceptually difficult to grasp for another participant p_1 . For example, p_0 may have built an EERG that expresses the problem $P = NP$ and directions for solutions. The given problem is a major open problem in theoretical computer science (Cook, 1971). It is not easy to understand the problem by merely defining it, even if you do so precisely.

Future EERG learning facilities could include the following.

1. Attach timestamps that express the creation time and date of nodes. This allows the newcomer p_1 , to follow the path that p_0 took in constructing the EERG, allowing p_1 to grow into the topic.
2. Allow participants to add improvements to the graph for the purpose of smoothening the way for newcomers.
3. These improvements could include ‘educational nodes’ – nodes that express information to train the concept to which the conceptual node is attached, for example the node that represents the problem $P = NP$.
4. Moreover, test-nodes could be added to determine the degree to which someone understands a given node. So, a test-node contains a test in the sense of an examination or challenge for participants. Here I see an additional possibility to apply the Orcoba-Approach in the context of real-time·collective·formal·thinking. Namely, by interpreting the educational nodes as composition-records and the tests as challenge-sets. This way the development of educational nodes could benefit from the advantages that efficient implementations of the Orcoba-Approach offer. Or-evohut is such a potentially efficient implementation.

5. These improvements should also include meta-information (e.g. something that expresses in a formal-language: ‘Recommendation: for easier understanding, node n_1 should be read after n_2 ’). This way they can be produced by a reasoner as a suitable ‘path’ through the graph for newcomers. These paths, moreover, could be customised for different types of participants, so that they support adaptive learning (Paramythis and Loidl-Reisinger, 2004) or personalised learning (Klašnja-Milićević et al., 2015), which is related to the more general notion adaptive hypermedia (Knutov et al., 2009).
6. (Deductive) algorithms could be applied to make the creation of educational nodes far more effective than ordinary learning material creation. For example, the explanation for a node that expresses a concept, could be partially automatically generated from educational nodes attached to sub-concepts. For example, the $P = NP$ problem requires the understanding of complexity classes (Hartmanis and Hopcroft, 1971; Arora and Barak, 2009) and Turing computability (Turing, 1936). The explanation could be generated from these concepts with a more atomic meaning. In this it would be ideal if a principle of compositionality would hold for educational materials. That is: good educational material for a concept is equal to a mathematical function of good educational material for the atomic subconcepts of the concept, and the structure that combines these atomic subconcepts into the main concept. I do not believe that this is generally the case. Undoubtedly, often holistic elements will play a role as well. However, for many concepts it will probably hold up to a certain extent.

Chapter III.6

The SWiFT class of games

III.6.1 Introduction

SWiFT (Semantic Web in Fast Translation¹) is a specialised extension of the Orcoba-Approach for the purpose of fostering formal-fluency. It takes the shape of a game in which participants have to translate a text from natural language to a formal-language. SWiFT is not one system, but a family (or class) of systems, containing a base design, and a number of variants and extensions: the *SWiFT-class-of-games*. (See section II.4.2 (p. 52) for terminology and motivations regarding system design principles covering the notions variants, extensions etc.) As such, SWiFT forms an important part of the answer to research question 2.2 (p. 143).

III.6.2 Related work

III.6.2.1 Games with a purpose

SWiFT is related to *Games with a purpose* (Ahn, 2006). Games with a purpose seduce people to volunteer enriching Internet-content with new representations of information that cannot be created automatically. This purpose is wrapped in appealing online games. Von Ahn metaphorically speaks of 'human computation'. In the OntoGames project, Siorpaes and Hepp (2008) have adopted the same approach for

¹The name includes the term 'Semantic Web', however, the scope is broader than the Semantic Web (see section III.2.7.2 (p. 145)). The scope is that of integration-of-automated-deduction. The term was, among other things, included due to my close relations with the Semantic Web research community, and the familiarity with the term by many.

creating Semantic Web content, and achieved promising results. A difference with SWiFT is that the focus of OntoGames is mass participation, with the disadvantage that the game must not be too difficult to play and some Semantic Web content authoring tasks must be sacrificed to make the game attractive for many. I assume that this limits the games to the creation of fairly lightweight content. Games within the SWiFT-class-of-games explicitly require training before participating. Some require much training, such as SWiFT-Full.

III.6.2.2 Translating natural language texts

In SWiFT games people are not writing down domain knowledge they already have, but are asked to translate a natural language text as faithfully as possible. This can result in particular difficulties caused by features of text and natural language, such as ambiguity and vagueness. To the best of my knowledge there is no research into guidelines for this specific goal.

III.6.2.3 Collaborative knowledge creation

The game enables participants to work together in creating formal representations. In the literature roughly two groups of systems are described with the same aim. The first group is formed by Semantic Wiki systems (such as SemanticMediaWiki²). Their main collaboration mechanisms are discussion pages (each content page has an associated discussion page). Rules or guidelines on content creation and collaboration are sometimes documented in separate pages (e.g. in FreeBase³, but these pages cannot be edited by users). BioMedGT is a Semantic Wiki that organises collaborative content creation through workflows. Different workflows exist for creating and changing a terminology.⁴ Proposed changes go through several states (proposal, accepted proposal etc.), and there are several user roles with different responsibilities within the proposal lifecycle (e.g. Expert, Curator).

The second group is formed by ontology engineering systems such as Collaborative Protégé⁵ and NeOn-Cicero⁶. Several elements of these systems are also present in SWiFT (such as voting mechanisms for adopting or rejecting changes). These systems, however, operate within the paradigm of top-down ontology development, and do not focus on the process of translating a given piece of information with high formal-fluency.

²<http://semantic-mediawiki.org/>

³http://www.freebase.com/view/en/content_guidelines_and_policies

⁴http://biomedgt.nci.nih.gov/wiki/index.php/Scenarios_Discussion

⁵http://protegewiki.stanford.edu/index.php/Collaborative_Protege

⁶http://www.neon-toolkit.org/component/option,com_wrapper/Itemid,128/

III.6.3 SWiFT-Base

The base design of SWiFT, SWiFT-Base, consists of those design ingredients of which all SWiFT designs in the rest of this work are either extensions or variants. It is defined as follows. First, it is a Game-Based-Orcoba, among other things to motivate participants and maximise accessibility of SWiFT to many, accelerating the evolution process. SWiFT proceeds, in brief, as follows:

- (1) *Translation-round*: Competing teams translate the same natural language text into a formal language, each player covering a fragment that does not overlap with other fragments. *translation-round*

- (2) *Question-attack-round*: Then teams challenge each other's translations by posing questions that have a very specific answer based on the content of the text. They do so without seeing the translation of the other teams. A question could for example be: 'What are all vitamins mentioned in the text that act as a catalyst for cell growth?' *question-attack-round*

- (3) *Algorithmic-defence-round*: For each question, each team, including the one that posed a given question, then does the following: *algorithmic-defence-round*
 - (a) *Algorithmic-construction-stage*: the team tries to construct a formal query ('deduction-algorithm') to automatically deduce the answer from their translation. *algorithmic-construction-stage*

 - (b) *Algorithmic-execution-stage*: the team executes the deduction-algorithm against their own translation. *algorithmic-execution-stage*

A short translation time, a low amount of effort needed to construct the query, and correct answers contribute positively to the final score of a team. Several extensions and variants of SWiFT are explored in this book.

In the context of SWiFT, the players described above are sometimes called fluency-players. This additional qualification allows these players to be distinguished from a second type of players: the players who develop new constitutions for formal-fluency. Note, in the terminology of the Orcoba-Approach, a fluency-player is equivalent to the composition-record-player. A treatment of a game for composition-record-players for SWiFT is beyond the scope of this book.

III.6.4 Validation of design decisions

III.6.4.1 The base metric for the quality of translations

The ideal of maximised deducibility forms the basis of the metric for quality of the translation in SWiFT-Base. In other words, the more sentences can be deduced from the translation by means of automated-reasoning, the better. This vague notion has been made precise with the metrics automatically-deducible-fraction and weighted-automatically-deducible-fraction in definition 36 (p. 179) and definition 37 (p. 179). The scoring procedure of SWiFT-Base is based on the automatically-deducible-fraction.

The automatically-deducible-fraction itself, however, has not been chosen as the scoring procedure. The problem is that it is in most cases not an operational-metric, i.e. there is no efficient procedure to determine its value. In fact, the scoring procedure in SWiFT-Base is an approximate-metric for the automatically-deducible-fraction. This is explained as follows. The fact that the questions are intended to ‘attack’ the other teams, creates a strong incentive to select queries (i.e. reasoning-goals) that the other teams may not be anticipating. This incentivises other teams to pursue maximisation of the automatically-deducible-fraction, because they do not know ‘what is coming’. If they, for example, limit their translation to a part of the text, they will achieve a low score when there are no or only a few questions about that part. The fact that each team also has to deduce the answer to the questions it poses from their own translation, creates a motive to choose questions that are spread out evenly over the different parts of the text. After all, as said before, the team is already motivated to spread out its translation-effort evenly over the text. Hence, questions it is able to answer will also have to be spread out evenly. All this suggest a positive correlation between the scoring procedure and the automatically-deducible-fraction.

It is important to note that the current design of the scoring procedure of SWiFT-Base does not keep into account the relative value people assign to questions: see section III.3.4.1 (p. 165) and in particular definition 24 (p. 166).

Moreover, the power of this scoring procedure is that it does not prescribe how the text should be translated. Instead it quantifies the way the translation should perform. In the literature, unfortunately, I predominantly encountered prescription-based methods with regard knowledge representation and reasoning and knowledge-engineering. Working with an adequately chosen operational-metric instead, separates the method from the scoring procedure. This allows a great level of creativity from the part of the players, including those that take the role of composition-record designers. They may come up with new ideas, allowing methods to evolve beyond what can be foreseen by a professional designer in the field of knowledge representation and reasoning. This philosophy of separation by means of metrics has been described in more detail in section III.3.1.1 (p. 152) and subsequent subsections.

III.6.4.2 Queries as algorithms

In the traditional view, a query is merely a ‘formal’ specification of a question, and a reasoner is the algorithm that answers that question. In SWiFT, however, queries for reasoners are also considered to be computer programs, or algorithms, in a very high programming language. The reason is that this allows a comparison to be made between *any* expression in *any* computer-language intended to specify a deduction. In other words, it unifies all these languages under one common notion.

This unified view is needed in some variants of SWiFT-Full. In the latter, one may use *any* sound algorithm to deduce the answers to questions. For example, instead of defining a desired deduction with the traditional ‘reasoner + query’ approach, one is allowed to define a dedicated deduction-algorithm in the programming language C++ that retrieves all constants for which a specific predicate hold from a first-order logic database. With the unifying perspective, it is possible to analyse both definitions in the same way, whether it be a ‘query’ or a ‘computer program’.

Some other philosophical reflections may further corroborate that the distinction between ‘queries’ and ‘algorithms’ is, from a certain perspective, imaginary. This perspective is not only useful within SWiFT, but also in a broader ontological sense. (1) From a purely computational perspective, both are parts of the definition of the total computation, or algorithm. A reasoner can be regarded as a higher order algorithm, which transforms the query into a dedicated algorithm that deduces a specific structure from the KR-base. In fact, the reasoner can be considered to be an interpreter or compiler, and the query a computer program in the language of this interpreter/compiler. (2) Queries often define patterns to be matched, which can be considered to be a (high level) definition of a computation. (3) Most of us are already used to calling some languages ‘programming languages’, while, with the conceptualisation within the reasoner/theorem proving world, they should at least partially be called ‘query languages’: declarative paradigms, such as functional, logic and constraint programming. This shows that both terms are already tacitly conceptualised as being interchangeable. (4) The converse of point 3 also holds: many query languages contain constructs that are identical to those found in programming languages, such as the ORDER BY construct in SQL (compare with a sort-function in many computer languages).

III.6.4.2.1 Queries as algorithms in SWiFT

Some more information about how this may be used in games that belong to the SWiFT-class-of-games may elucidate the importance of this view. In a SWiFT-game-design the quality of a translation may be partially determined by the average

syntactical complexity⁷ of the deduction-algorithm. A higher complexity of the deduction-algorithm contributes negatively to the score of the players who created the translation. The reason for this design decision is that, given a translation T , the simplest deduction-algorithm that can be created to deduce the answer to question Q from T , the better T is considered to be with respect to that question. With the unified perspective (1) syntactical complexity measures for source code complexity can be applied to what would traditionally be called ‘queries’, and (2) the syntactical complexity of any computer-language expression that (helps to) define a deduction can be compared to others, whether it is, for example, a program in C++ or a query in SPARQL.

Note that comparing the complexity of the constructed queries, in fact, involves more than this. The chosen computer-language will in most cases have an influence on the minimal complexity of the deduction-algorithm. For example, for many types of questions, the deduction-algorithm in the programming language C will be more complex than in Racer (Haarslev et al., 2012). This is because Racer, as a dedicated OWL-reasoner, offers efficient abstractions for specific forms of automated-deduction and predefined code that supports this reasoning. In other words, a great part of the burden of the automated-deduction strategy is carried by Racer, and not by the deduction-algorithm. In C, a general purpose programming language, the abstractions and code have to be manually implemented in the deduction-algorithm. This makes it a challenge to compare deduction-algorithms. A direction for a solution is to consider the computer-language in combination with the deduction-algorithm as the complete algorithm. For example, the parts of the source code of the interpreter or compiler of the computer language that are used for the deduction-algorithm and the deduction-algorithm itself are analysed for their combined syntactical complexity. How to do this adequately is an open issue.

III.6.5 Extensions and variants

The following briefly introduces extensions and variants of SWiFT-Base. An overview of these can be found in fig. III.6.1 (p. 224).

III.6.5.1 SWiFT-Full

SWiFT-Full is a subclass of SWiFT-Base that offers the players a challenge that is very close to testing full mastery of formal-fluency. (Or, in terms of section II.4.5.1 (p. 57): it offers a challenge-set that is (close to) ideal for the capability of formal-fluency.) It is defined by the following additions to SWiFT-Base: (1) the players are free to choose

⁷Not to be confused with the computational complexity.

any formal-language they want to represent the natural language texts, (2) the players may choose any (sound) automated-deductive-reasoners to deduce their result and (3) there are no restrictions in the selection of natural language texts that are offered to the players. This implies that players may choose different formal-languages and different automated-deductive-reasoners, or even develop these partially or fully themselves. Note that this explains the word ‘full’ in SWiFT-Full. However, it has some major disadvantages, and therefore this work includes complementary SWiFT variants and extensions that deal with these.

III.6.5.2 Sub-research-questions

An essential aspect of any SWiFT is to incentivise people to train their formal-fluency. This leads to the following question.

Research question 2.2.1. *[Incentives SWiFT-game] Which incentives for participating and improving formal-fluency does SWiFT provide, and how could the game be improved to make these incentives stronger?*

[Conclusion(s): page 240 and page 297] ■

Another set of research questions is inspired by disadvantages of SWiFT-Full. These disadvantages are:

1. In SWiFT-Full it is quite difficult to compare the results of different teams, because they may have used different formal-target-languages and quite different specifications of a deduction-algorithm.
2. The freedom players have in choosing their own formal-target-languages also comes with several burdens for these players. They have to select, install and/or develop the necessary software tools, such as automated-deductive-reasoners and the deduction-algorithms. They have to balance the difficult trade-off between expressivity and computability while selecting a suitable formal-target-language.
3. Confronting players immediately with full-blown formal-fluency can be intimidating, and therefore discouraging.

These disadvantages inspired the following corresponding research questions:

Research question 2.2.2. *[Easier comparison translations] How to design a SWiFT-game or variant thereof in which the quality of the translation of the players is easier to compare than in SWiFT-Full?*

[Conclusion(s): page 242 and page 297] ■

Research question 2.2.3. [No immediate full-blown formal-fluency needed] How to design a SWiFT-game in which the player is not immediately confronted with acquiring full-blown formal-fluency, but which still has many to most of the desired effects of SWiFT-Full?

[Conclusion(s): page 242 and page 298] ■

Research question 2.2.4. [Reduced human-resources] How to design a SWiFT-game or variant thereof that is effective for fostering formal-fluency, but that is less human-resource intensive than SWiFT-Full?

[Conclusion(s): page 298] ■

Another aspect is the scoring procedure of a given SWiFT design. The SWiFT-Base design does not uniquely define a scoring procedure. It allows different implementations. *Example: How do you compare translations if they both yield the same correct result for a given question, but where it is – in some respect – more difficult to deduce it from the first translation?* Moreover, the question is whether the scoring procedure of SWiFT-Base or a specific implementation of it, corresponds with what one would perceive as a high quality translation in possibly other respects. This yields the following question.

Research question 2.2.5. [Effective scoring procedure] Is the scoring procedure effective in the sense that it actually exposes translation weaknesses and strengths?

[Conclusion(s): page 243] ■

III.6.5.3 Complementary approach

I believe that most disadvantages mentioned in section III.6.5.2 on the previous page should not be overcome by replacing SWiFT-Full, but by complementing it. SWiFT-Full has the potential to become an excellent instrument to foster full-blown formal-fluency. This is part of one of the main research questions of this study. If SWiFT-Full is complemented with variants of SWiFT that address the aforementioned disadvantages by sacrificing some of the scope of SWiFT-Full, a full suite of instruments to train formal-fluency can be achieved. The variants developed during this study are SWiFT-Fixtal and SWiFT-Focused.

III.6.5.4 Iterative approach

Another reason to design and implement variants of SWiFT-Full is that it is an iterative approach for the research and development of SWiFT-Full. Many conclusions drawn

about these variants will also hold for SWiFT-Full. Some variants, however, may be less resource intensive to develop. This enables the researcher or developer to collect insights about the functioning of SWiFT-Full and how to improve it more efficiently. In this study, such generalised conclusions from variants will indeed be made.

More generally, experiences with specific subclasses of the SWiFT-class-of-games may be generalisable to wider classes within the SWiFT-class-of-games.

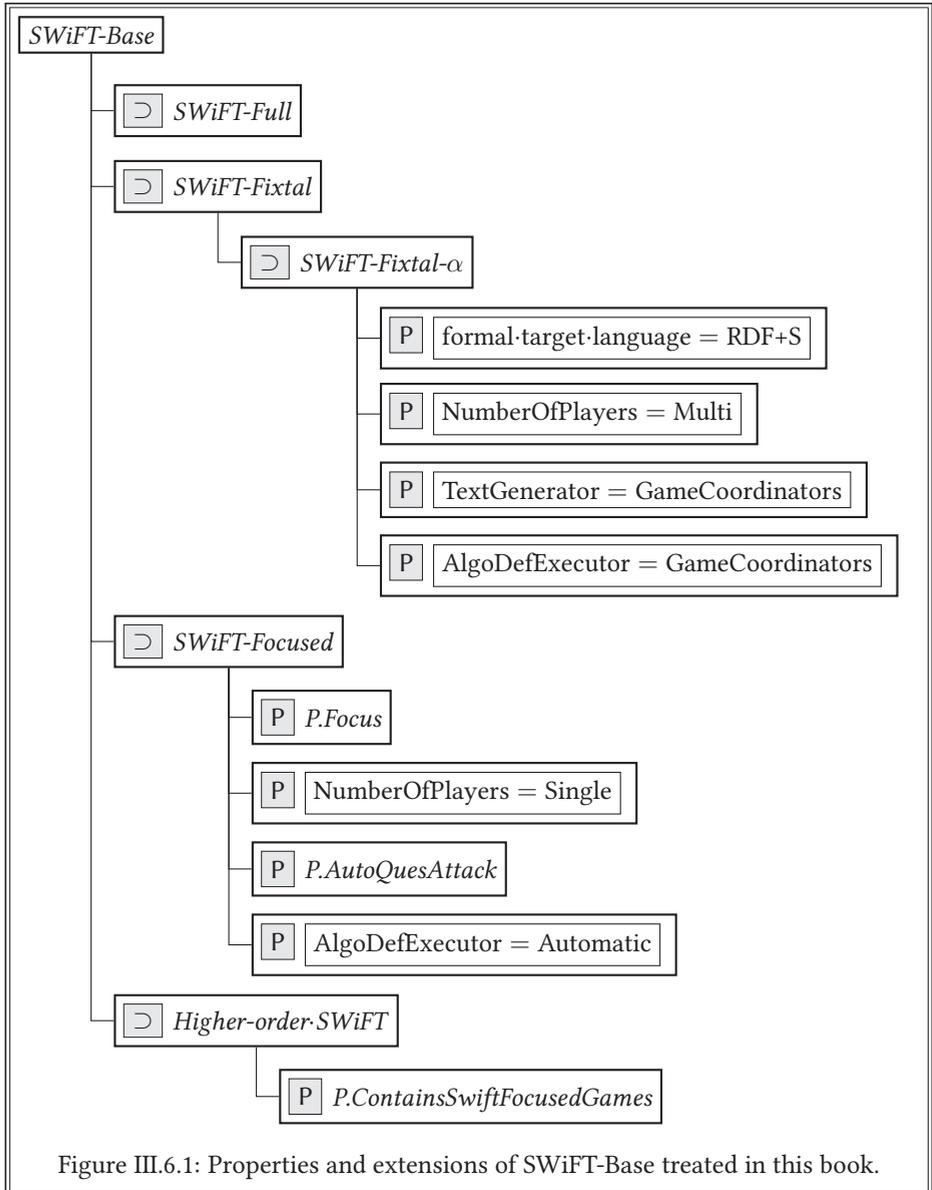
III.6.5.5 SWiFT-Fixtal

SWiFT with Fixed target language (SWiFT-Fixtal) is the subclass of SWiFT-Base within which the structural-semantic-conditions (see section III.3.5.5 (p. 184)) of the formal-language are fixed. This means that the structural-semantic-conditions of the formal-target-language may not be chosen by the players of the game. It has been developed in response to the research questions posed in section III.6.5.2 (p. 221). SWiFT-Fixtal- α is the game that was object of study during the first experiment with SWiFT. It is a member of the SWiFT-Fixtal class of games, and therefore, has a fixed target language. SWiFT-Fixtal and SWiFT-Fixtal- α are described in detail in chapter III.7 (p. 225). Experimentation with it is described in the chapter IV.3 (p. 323).

III.6.5.6 Higher-order-SWiFT and SWiFT-Focused

In higher-order-SWiFT the complete challenge of mastering formal-fluency is broken up into smaller subcapabilities. Each subcapability is addressed in a short subgame: a SWiFT-Focused game. The SWiFT-Focused games are presented to the player in order of a gradually increasing level of difficulty. Each SWiFT-Focused game is a direct instance of the SWiFT-class-of-games, while higher-order-SWiFT is a subclass of higher-order-Orcoba. SWiFT-Focused is covered in chapter III.8 (p. 245).

Higher-order-SWiFT and SWiFT-Focused have been developed in response to the research questions in section III.6.5.2 (p. 221) and research questions raised by SWiFT-Fixtal.



Chapter III.7

SWiFT-Fixtal and SWiFT-Fixtal- α

(Before starting this chapter, I would like to gently remind the reader of the call for feedback on page i.)

III.7.1 Introduction

SWiFT with Fixed target language (*SWiFT-Fixtal*) is the subclass of SWiFT-Base in which the structural-semantic-conditions of the formal-language are fixed. This means that the structural-semantic-conditions may not be chosen by the players of the game, in contrast with SWiFT-Full. For the sake of simplicity, in the context of SWiFT-Fixtal the notion ‘fixed formal-target-language’ is synonymous to ‘fixed structural-semantic-conditions’. The reader, however, has to keep in mind that the domain-related-semantic-conditions may be extended during the SWiFT-Fixtal game. SWiFT-Fixtal has been developed in response to the research questions posed in section III.6.5.2 (p. 221). *SWiFT-Fixtal- α* is an instance (specific implementation) of SWiFT-Fixtal. SWiFT-Fixtal- α is the game that was object of study during the first experiment with SWiFT. It is a member of the SWiFT-Fixtal class of games, and therefore, has a fixed target language. SWiFT-Fixtal and SWiFT-Fixtal- α are described in detail in chapter III.7. Experimentation with SWiFT-Fixtal- α is described in chapter IV.3 (p. 323).

Important reasons for having a fixed formal-target-language is that it provides a solution to the research questions posed in section III.6.5.2 (p. 221):

1. The results of adversaries are easier to compare, which is more complicated if the formal-target-languages of the adversaries differ, providing a response to research question 2.2.2 (p. 221). More details about this can be found in section III.6.4.2.1 (p. 219). A particular challenge-assessment is presented in section III.7.4.2 (p. 238).
2. Self-evidently, with the fixed formal-target-language the players do not have to select, install and/or develop the necessary software tools, such as automated-deductive-reasoners. They do not have to balance the difficult trade-off between expressivity and computability while selecting a suitable formal-target-language. This is a response to research question 2.2.4 (p. 222).
3. By fixing the formal-target-language, the game designer can expose the players to simpler forms of formal-fluency that are less demanding than full-blown formal-fluency. Typically this can be achieved by selecting a suitable less expressive formal-target-language. This is a partial response to research question 2.2.3 (p. 222).

It is important to note that SWiFT-Fixtal is *not* intended to replace SWiFT-Full. It is intended to complement it, as has been explained in section III.6.5.3 (p. 222).

This chapter presents the design of SWiFT-Fixtal- α , a specific instance and implementation of SWiFT-Fixtal. An experiment with the game is covered in another chapter: chapter IV.3 (p. 323).

The study of SWiFT-Fixtal- α is also intended to draw conclusions about the functioning of SWiFT-Full and develop ideas to improve it, and other games within the SWiFT-class-of-games, as explained in section III.6.5.4 (p. 222).

III.7.2 Formal-target-language and reasoners

The formal-target-language of SWiFT-Fixtal- α is formed by combining RDF and RDFS (see section III.2.4.1 (p. 141)). (In the sequel, the combination is abbreviated with RDF+S. In other words, RDF+S = RDFS + RDF.) The advantages of choosing RDF+S is that (1) there exists reasoner support for it (such as Jena and Sesame) (2) it is the simplest standardised set of semantic conditions available now, therefore, the players do not have to make too big a conceptual leap at once. The main disadvantage, of course, is that with its very limited expressivity, deducibility will also be limited. However, since both teams have exactly the same disadvantage, their results are nevertheless comparable. Note that this decision leads to another one: questions that need a greater expressivity than RDF+S provides are disqualified. *Example: One may not expect a team to deduce the answer to the question “Is Carlo a woman?” from*

the translation of a text that merely states that men and women are disjoint sets, and Carlo is stated to be a man. The reason is that one cannot express disjointness of sets in RDF+S such that it enables reasoners to make deductions about disjointness of sets. Note that RDF+S has a clearly defined set of formal semantic conditions (Hayes, 2014). It is not just a ‘semi-formal-language’, as some seem to believe. However, there is a caveat: the standard existing reasoning support for RDF+S is based on the query language SPARQL (W3C SPARQL Working Group, 2013). I argue that the soundness of SPARQL cannot be fully proven based on the formal semantics of RDF+S. Nevertheless, SPARQL can still serve as a starting point for acquiring many of the skills necessary for working with systems with sound inference engines such as OWL. Note that when translating natural language both RDF and RDFS are applicable. A text may contain schema and ontology level information such as “Vitamins are organic compounds.”, which is a subclass relation that can be expressed in RDFS.

III.7.3 The composition-record

This section presents the initial composition-record I developed for SWiFT-Fixtal- α . First, section III.7.3.1 presents an overview. Then preliminary definitions concerning the knowledge representation follow. These are needed in section III.7.3.2.3 (p. 232) to present the part of the composition-record that are in the focus of this chapter: a selection of guidelines for node definitions.

III.7.3.1 Overview of the composition-record

The composition-record consists of two parts: a human part and a complementary artefact part (see section II.3.1 (p. 44)). An overview and explanation follow.

III.7.3.1.1 Human part

The human part of the composition-record is organised in a stack of layers, of which each layer forms the chosen concretisation of the more essential, but less operational, definitions in the preceding layer. Note the distinction between the rules of the game and the rules of the composition-record. The latter are described in this section, even though some rules of the composition-record are copies of rules of the game. The advantage is that this way the essential elements as well as the concrete elements of the complete composition-record are made explicit. The lower layers are closer to the essential, fundamental, conditions, and are expected to evolve slowly, while the higher layers are closer to the realisation of these ideas. They are expected to

be evolve quickly because there may be many different ways to realise the essential conditions.

The current composition-record is subdivided into three layers. The *ground layer* contains two rules: (1) The composition-record may not violate the rules of SWIFT. (2) The principle goal is to translate the given text into the formal-target-language, with as big an automatically-deducible-fraction as possible (see section III.6.4.1 (p. 218)), and in as short a time as possible (*end of list*). The *second layer* contains in total 6 articles that can be divided into the following two main categories: (1) knowledge representation conditions, and (2) the language coordination protocol.

The *knowledge representation conditions* described in the second layer include: *SWiFT·Turtle* (1) The formal-target-language for the translation is *SWiFT·Turtle*, which is basically RDF+S written in Turtle (Becket David, 2011) with some modifications. (2) Each node must be accompanied with a definition in natural language that must satisfy certain conditions. The purpose of these conditions is to indirectly contribute to a greater automatically-deducible-fraction. These conditions and their motivation are explained in section III.7.3.2.3 (p. 232).

The third layer of the composition-record contains a complete specification of the *SWiFT·Turtle* language.

The game rules allow only communication between players that pertains to the language coordination protocol, and prohibits communicating the content of the text-fragment being translated (which is motivated in section III.7.4 (p. 235)). The implementation of the coordination is as follows. To tackle one challenge at a time, the coordination is restricted to the definition of the nodes in this version of the composition-record. Higher-order coordination activities, such as standardisation of multi-node representations, are left as future work. There is a shared KR-base that contains the definition of all nodes created by the players. Before a player may write down a *SWiFT·Turtle* statement, the player and his team must first go through the coordination procedure displayed in fig. III.7.1 on the next page for each node the statement contains. The figure is slightly simplified. In the actual game, among other things, players may revise their opinions in a later stage (giving space to growing experience) and explain the reason for declining using voice chat.

III.7.3.1.2 Artefact part

The artefact part of the composition-record supports the node definition and coordination process. It is coined *Constitution-Based·Subleme* (CBS). I developed it as an extension to the existing Subleme Wiki system¹ by Michiel Visser. A relevant part of its functioning is explained in fig. III.7.2 (p. 230). In the current setting users do not

¹<http://code.google.com/p/subleme/>

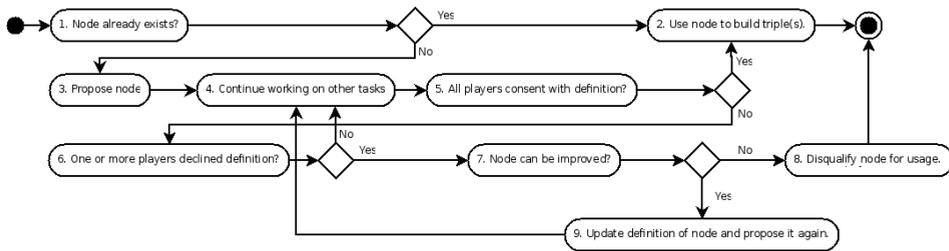


Figure III.7.1: UML activity diagram of the core element in the language coordination process: proposing new nodes. Figures below contain screenshots of the tool support provided with some of these activities.

express RDF+S in Constitution-Based-Subleme, rather they write them down in Word in the track-changes mode. For communicating about declined definitions voice chat (Skype) is used. In a future version of CBS it will be possible to use discussion pages. For the purpose of the algorithmic-defence-round, the RDF+S inference engine Sesame has been integrated into CBS. This allows players to easily construct and apply queries to their translations.

III.7.3.2 Formal-target-language

The formal-target-language is built upon the Exact-Entity-Relation-Graphs I developed in the context of System- α -for-real-time-collective-formal-thinking, see section III.5.4.1 (p. 202).

It consists of two parts. The first is the formal-target-language, SWiFT-Turtle, which is semantically consistent with RDF+S (Hayes, 2014) and has a syntax similar to Turtle (Becket David, 2011). Hence, in contrast with Exact-Entity-Relation-Graphs, it has a mathematically defined semantics.

For the purpose of the game, SWiFT-Turtle has a simplified syntax in comparison to Turtle. Moreover, SWiFT-Turtle makes explicit a semantic condition pertaining to the nature of the entities that is not made so in RDF+S (see article 4 (p. 234)). Therefore, every SWiFT-Turtle document can be confidently translated into valid RDF+S, while the opposite is not guaranteed.

The second part are conditions which regulate the way the vocabulary of SWiFT-Turtle may be extended with new nodes. In contrast to the rigorous definition of the *predefined* part of the semantics of RDF+S (Hayes, 2014), support for this process is hardly present, as indicated in related work. I believe that a transition from natural language to the formal-language is unavoidable in this process, and if its difficulty is not recognised, it can lead to serious problems. This transition

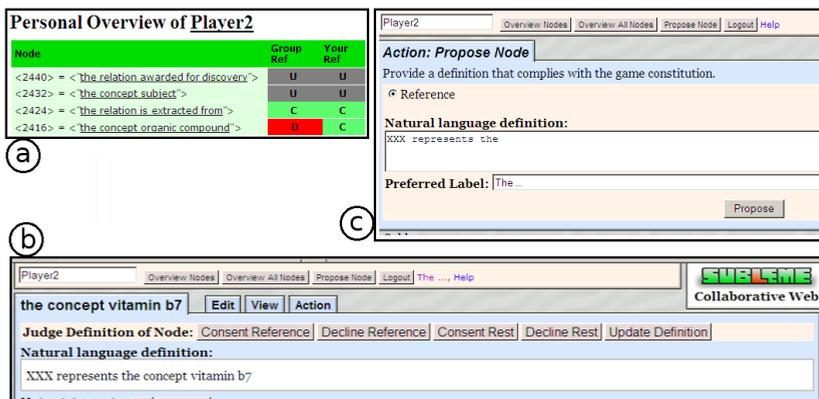


Figure III.7.2: A selection of screenshots from Constitution Based Subleme. (a) The *node overview*. This shows the labels of all nodes, and current group voting status ('group'), and the personal voting status ('your') is shown per node: C(onsented), D(eclined), U(ndetermined). Before a decision can be made, a player first has to click a node, which brings the node into focus. (b) A *node in focus*. This shows the definition of the node, and allows it to be improved or a vote to be casted. (c) *Proposing a new node*.

normally takes place completely unregulated in RDF+S by allowing people to attach any natural language label to define the meaning of a node. Typically, this happens by including names pertaining to the referenced entity in the URI, for example <http://example.org/violin>. An attempt to fill this gap is made in section III.7.3.2.2 (p. 232) and subsequent sections.

III.7.3.2.1 SWiFT·Turtle

SWiFT·Turtle is a graph-based KR-language, in which sentences consist of triples of nodes, expressing a relation (the node in the middle, acting as an edge in the sense of graph theory) between two other entities (the two outer nodes, acting as vertices as we know them from graph theory). SWiFT·Turtle can be considered to be a cross between Exact·Entity·Relation·Graphs and RDF+S. EERGs were introduced in section III.5.4.1 (p. 202), and RDF+S in section III.2.4.1 (p. 141). In the following, SWiFT·Turtle is defined by comparison with the other two languages.

A comparison of the syntax of SWiFT·Turtles and EERGs are as indicated in table III.7.1 on the facing page and in the tables below:

Table III.7.1: Comparison SWiFT·Turtle and Exact·Entity·Relation·Graph.

Topic	SWiFT·Turtle	Exact·Entity·Relation·Graph
Notation for nodes	<56>	{56}
Digital entity	"ten"	{"ten"}
Preferred label	<"Anish Giri">	{ <i>Anish Giri</i> }

Human definition ('Human description' in chapter III.5 (p. 197))	
SWiFT·Turtle	<<"■ represents the set of natural numbers.">>
EERG	{"■ represents the set of natural numbers."}

Triple	
SWiFT·Turtle	<"Anish Giri"> <"has nationality"> <"the Dutch nationality">.
EERG	{ <i>Anish Giri</i> —{ <i>has nationality</i> }->{ <i>the Dutch nationality</i> }}

A comparison between SWiFT·Turtle and Turtle is as follows. Their syntax and semantics is the same, with a few exceptions, explained and motivated in the following. The Uniform Resource Identifiers ("URIs") in SWiFT·Turtle are limited to integers, for example <465> could be introduced to represent the concept vitamin A. These identifiers are chosen because they are neutral, instead of containing natural language. In this way we can regulate the transition from natural language to the formal-language. (More about this will follow in section III.7.3.2.2 on the following page.) A human definition is expressed in a string-literal, and then attached to a node with a predefined relation for this purpose: the node <1>. An example is

"represents the concept vitamin A" <1> <33>.,

which attaches a human definition to the node <33>. (Note that the period after each triple is part of the SWiFT·Turtle and Turtle syntax.) For this, the convenience notation <"represents the concept vitamin A"> is used, which means: the node to which the given human definition is attached. In the latter example, this is the node <33>. The human definition must follow very specific guidelines that will be explained later.

Moreover, purely as a mnemonic aid, a preferred label may be added to the node using the predefined relation <2>. An example is

"vitamin A" <2> <33>.

In this book, however, the convenient notation <<"■ represents the vitamin A.">> is used. Consequently, the vocabulary of SWiFT·Turtle includes the complete vocab-

ulary of RDF+S and extends it with two relations, to attach preferred labels and natural language definitions to nodes. For the convenience of the players, the prefix ‘xsd’ may be omitted when indicating XML datatypes.

III.7.3.2.2 Transition from natural language to formal-language

As stated before, I argue that most people are prone to suboptimal and erroneous modelling decisions because they have to make a transition from natural language to a formal-language. This transition takes place on two levels. First, the text, worded in natural language, has to be translated into a formal-language. Second, expressing real world knowledge in a formal-language, requires the formal-language to be extended on the fly, in the case of SWiFT-Turtle with new nodes. The meaning of these new nodes is most obviously coordinated in a language that the participants already share: natural language. In fact, the notions to be introduced in the formal-language were shaped in natural language, causing the transition to be inherent to the process of expressing information in a formal-language.

If the cognitive difficulty of this transition is not well-recognised, translations can easily become faulty. Using natural language people are able to interpret language expressions using common sense knowledge and using their estimation of the context in which the expressions are placed. The previous experiments, among other things with System-Real- α , showed that people are inclined to continue using this natural language style of interpretation, even when expressing information in a formal representation with a predefined, and thus deviating semantics (Groenouwe and Top, 2008).

The guidelines to prevent these mistakes are essential to the approach. A selection of the most important ones follow. For illustrative purposes, each guideline starts with a natural language fragment that could tempt the translator to violate it.

III.7.3.2.3 Conditions node definitions

Article 1 (The definition of each node must form a fixed reference to an entity). *The definition of each node must form a fixed reference to an entity. It is tempting to translate “In the Irish general election of 2016, some people voted for the Green Party” literally by creating the node <<“some people”>> for the part of the expression the contains “some people”. However, the implication of this translation is someone reusing the same node in the translation of “In the Irish general election of 2016, some people voted for the DUP”. Common sense dictates that the people in these two sentences are not the same, because each person can only vote for one party at any given election. Nevertheless, the formal semantic conditions of RDF demand that they refer to the same entity. In this case, an RDF reasoner will deduce “In the Irish general election of 2016,*

some people voted for both the DUP and the Green Party”, which is clearly incorrect. To prevent translators from making this kind of mistake the guideline states:

Exclusively use definitions that refer to one and the same entity regardless of the statements in which they are used.

Note that the notion ‘entity’ has to be interpreted broadly. It also includes abstract (mental) entities, such as the <<"the class of food products">>, or <<"the frequency range 10Hz to 20Hz">> and a set of entities is also an entity, for example <<"the researchers of team X300 of company Bionova in 2008">>.

A peculiar variant of this mistake has occurred in FreeBase²: users have specified in their profile the concept ‘Nothing’ as value for the property ‘hasSpouse’. Wrong inferences will result from that statement: all people that are trying to state that they are single are actually stating that they all share one and the same spouse with the name ‘Nothing’. The problem here is that nothing does not refer to any entity and thus violates this article. ■

Article 2 (The blacksquare notation). *The probability of violating article 1 on the preceding page may be reduced by introducing an auxiliary article. Every definition should start with the text “■ represents the ...”. The black square refers to the node the meaning of which is being defined. The black square (■) stresses that it is about a (mental or physical) entity, and the word ‘the’ stresses that this entity must be fixed. For example, it may alert a player that “some food products” cannot be used, because following the new rule, this would lead to <<"■ represents the some food products.">> which is clearly grammatically incorrect English.*

For brevity, this chapter suffices with using preferred labels in the triples if it is clear to the reader what entity is being referred to. In preferred labels the blacksquare notation is not required, for example <<"the team X300 of Bionova in 2017">>. ■

Note that the predecessor of this article is part of System-Real- α , see section III.5.4.1.2 (p. 203).

Article 3 (Definitions must be univocal). *“Akwasi used the chromatograph in his experiment”. One could be tempted to define <<"■ represents the chromatograph.">> for the node representing the chromatograph referred to in the sentence. However, in the context(s) described by the complete text, it may be unclear to which chromatograph this node refers. ■*

Note that the predecessor of this article is part of System-Real- α . See article 1 (p. 206).

²<https://en.wikipedia.org/wiki/Freebase>

Article 4 (Avoid extension/intension ambiguities in definitions). “The wavelength of ultraviolet light is 200 nm”. A faulty translation would be: <“The wavelength of ultraviolet light”> <“is equal to”> <“The length 200nm”>. The problem lies in the first definition. Some definitions, such as this one, have a so-called intensional and extensional interpretation (Carnap, 1956). The extension of a definition in natural language consists of the entity or entities it points out, while its intension is formed by the associated qualities/properties the definition expresses about these entities. In the given problematic definition, the extension is a distance, while its intension relates to the property of being the wavelength of ultraviolet light. This can lead to errors as the following. Suppose one additionally states:

<“The wavelength of ultraviolet light”> <type> <“The concept thing that is difficult to measure”>.

In combination with the first statement, one can deduce:

<“The length 200 nm”> <type> <“The concept thing that is difficult to measure”>.

which is obviously not what was intended. Such ambiguities must be avoided, and the definition should for example be replaced with a definition that solely refers to the intension or the extension of the previous definition, depending on what you wanted to express. ■

Article 5 (Definitions must be unique references to entities). “The scientist A. Doisy discovered vitamin K. [...] K. Funk was a scientific researcher, who ...”. This could lead to players adding both the nodes <<“■ represents the concept scientist.”>> and <<“■ represents the concept scientific researcher.”>>. This, however, is not allowed. Using the same node for the same entity implies that all inferences that involve that entity can be made (much) more efficiently than using several nodes. In the latter case you would need to include each synonymous node in your deduction algorithm.

Note that this is not a requirement of RDF in general, there is no ‘unique name assumption’. However, unique identifiers on the Web are also the most desirable situation. An important reason for deviating from this situation in RDF is that the full Semantic Web is so big, that it is infeasible to ensure yourself that a node has not been defined as yet. The situation in the game is different, and thus, the ground for allowing multiple nodes to denote the same entity becomes invalid. ■

III.7.3.2.4 Trade-offs

Some design decisions in the articles above may be in need of some additional explanations. They are related to the reflection on formal knowledge representation

in section III.3.5.10 (p. 190). First, the articles do not impose restrictions on how to negotiate the trade-off between quality and speed. Each player is free to determine the degree of granularity in representing a piece of information. (See section III.3.5.10 (p. 190)). A lower level of granularity, may contribute to longer node definitions. Players are free to further granularise their representation such that it consists of more elementary-nodes with simpler definitions.

Second, note that it may occur that the same piece of information is represented both formally and informally: information expressed in the natural language definition of a node, may occur again in terms of SWiFT-Turtle statements. When reading a natural language definition, it is important that the reader keeps this in mind. Otherwise he may erroneously assume that the information in the definition is not accessible for automated-deductive-reasoners. For example, a node N may well be defined in the following way: <<"■ represents the scientist with the name K. Funk.">>, and to this node may be attached: N <type> <"the concept scientist">. The reason is that in the current version of the composition-record, players first negotiate the meaning of URI in natural language, before they may use it to build triples. This allows the mentioned transition to take place as consciously as possible. After using it, (s)he may want to express information such that it can be exploited by deduction-algorithms, and represent part of the information present in the natural language definition *again*. Future versions of the composition-record will support that part of the meaning of a URI is *defined* using the formalism.

III.7.4 The game design

This section describes and motivates the game design as it currently stands in more detail.

III.7.4.1 Game phases

SWiFT-Fixtal- α follows the SWiFT-Base design as described in section III.6.3 (p. 217), and adds some details to it. The *game-coordinator* is a person who coordinates certain game activities. This person is not a player, and the rest of this section includes his or her activities. The game consist of the following phases: *game-coordinator*



Each phase is now described in more detail. (Section IV.3.1.1 (p. 323) presents the same list of phases, but supplements it with examples taken from the experimental session.)

Training. The players thoroughly study the rules of the game and the composition-record.

Team formation. Two or more teams of equal size are formed by the game-coordinator.

Text assignment. During the text-assignment the game-coordinator divides a text into as many non-overlapping fragments as there are players in a team. Each player of each team then receives a fragment that is different from that of his peers. Consequently, each team covers the complete text. Players do not see the fragments of their team members.

Motivation. As stated before players have to translate a written text instead of their own thoughts. This has two operational benefits. First, it offers a frame of reference to compare the performance of different teams, and therefore, measure their progress. Second, it allows us to steer the process towards certain modelling difficulties. It is essential that the fragment for each player is part of a single, coherent text for the team. The questions to be asked later on require the players to build an integrated model for the entire text, given their individual translations

Translation round. The teams translate the text into a SWiFT·Turtle representation. They follow the guidelines prescribed by the composition-record as much as possible, including the consensus strategy.

Question attack round. The teams challenge each other's translations by posing questions that have a specific answer based on the content of the text. In this stage, all team members get access to the full text. Moreover, a question is legitimate if its answer can be (1) determined solely with the information provided in the text, and (2) deduced given the expressiveness of the formal-language and the language to construct deduction-algorithms, in this case SWiFT·Turtle and SPARQL.

Motivation. The combination of the fact the players do not see each other's fragments, and the questions being about the text *as a whole* has the following advantages. First, it pushes the players towards creating *reusable* representations, because they have to keep into account any possible sensible extension of their fragment. Second, it approaches the situation of integration-of-automated-deduction in larger communities. Most of these communities will be heterogeneous with people globally dispersed,

such as scientists, each expressing information about a local project. These people will not have the time to read the work of all others. Nevertheless, want the *aggregate* of their individual contributions to have the greatest automatically-deducible-fraction as possible. For this purpose, they need to find a way to mutually coordinate their language, while minimising the need of reading each other's work. In the game this has been simplified to only allowing language coordination, and no content exchange. In the current composition-record the only language coordination that takes place is on the level of node definitions. The fact that the questions are intended to 'attack' the other teams, creates a strong incentive to make the questions as difficult as possible, and thus, the test of quality as good as possible.

Algorithmic-defence-round. Each team then attempts to construct a deduction-algorithm that deduces the correct answer from their translation. The deduction-algorithm should not contain other information than the information provided in the question. In the present version of the game, the language to construct the deduction-algorithms is SPARQL, consequently the deduction-algorithms take the form of SPARQL queries.

Motivation. In combination with the question attack round, the algorithmic-defence-round forms a natural way of assessing the quality of the translations, because it coincides with the ultimate purpose of Semantic Web representations: effectively extracting answers to unforeseen questions from them through deduction-algorithms.

Determining the score. The score consists of the following 2 components:

(1) Time. The total times T_1 and T_2 teams 1 and 2 respectively needed to make the translation is determined (in minutes).

(2) Performance Algorithmic Defence. For each question Q_i , the points A_{1_i} and A_{2_i} the teams 1 and 2 respectively receive are both 0 when they did not correctly answer the question. Otherwise, 1 point is to be divided among both teams. If a team is the only one that was capable of answering the question, it receives the full point. In the other case, the portions they each receive of the point depends on the complexity of their deduction-algorithms (L_{1_i} and L_{2_i}). The portions have a ratio that is the inverse of the ratio of the complexity of their deduction-algorithms, or in formula form:

$$A_{1_i} = \frac{L_{2_i}}{L_{1_i} + L_{2_i}},$$

and

$$A_{2_i} = \frac{L_{1_i}}{L_{1_i} + L_{2_i}}.$$

This tallies with the intuition that the more complex your deduction-algorithm is

in comparison with that of your adversary the smaller the portion you receive of the point should be. The remaining question is how to determine the complexity of the deduction-algorithms. For this purpose, this experiment uses the Halstead metric (see section III.7.4.2). The overall question defence score $QD1$ and $QD2$ for the respective teams are defined as follows:

$$QD1 = \sum_{i=1}^N A1_i,$$

and

$$QD2 = \sum_{i=1}^N A2_i,$$

where N is the number of questions used during the question attack. This formula implies that all questions carry equal weight in determining this part of the score.

These two components allow the determination of the overall score. These scores $S1$ and $S2$ for the corresponding teams are:

$$S1 = \frac{QD1}{T1}C,$$

and

$$S2 = \frac{QD2}{T2}C.$$

where C is solely included for presentation purposes (to avoid scores that are small fractions). In the experiment $C = 1000$. The division by T intuitively translates to: an $n\%$ better score on the question defence must be realised in less than $n\%$ more time to get a higher overall score. This forces players towards formal-fluency.

Motivation. The motivation is already provided in the previous text.

III.7.4.2 Measuring the quality of translations

In the game a part of the metric consists of the amount of effort it takes humans to create a deduction-algorithm that answers the competency question. In SWiFT-Fixtal- α the deduction-algorithm is a SPARQL query defined for the question in the algorithmic-defence-round. A more complex deduction-algorithm corresponds with a lower quality of the translation. I use Halstead's *program length* metric known from the field of software complexity. Program length is defined as the sum of all

occurrences of *operators* and *operands* (Halstead, 1977). In SWiFT-Fixtal- α the choice for this metric is motivated by a study of (Bowen et al., 2006). This study measures the complexity of a representation in terms of the complexity of queries on the representation, just as in SWiFT-Fixtal- α . It established that in this context, more complex Halstead metrics such as *difficulty level* are highly correlated with the simple Halstead metric used in SWiFT-Fixtal- α .

Below is an example where all operands and operators are made italic. Notice that the choice whether a symbol is an operand or operator is not important to measure program length.

```
select ?location where
{ ?rel <synthesized_substance> <the_concept_vitamin_d> .
  ?rel <organ_location> ?location .
}
```

The program length of this query is 8 (we also count the *select...where* statement as an operator).

III.7.5 Remedies violations node definitions based on experiment

The remedies against the following possible causes for node-definition violations that occurred during the experiment is extending the composition-record with a player-friendly rewording of the analysis given in section IV.3.2.3 (p. 327).

- Deceiving natural language subjects
- Lack of awareness of the metaclass/superclass ambiguity
- Reference by property
- Expressing different names of the same entity

For example, the text I suggest adding to prevent univocality-violations caused by a lack of awareness of the metaclass/superclass ambiguity, reads as follows:

Article 6 (Avoid metaclass/superclass ambiguities). “*Vitamin A is a vitamin, and retinol is a type of vitamin A*”. It would be tempting to define the second occurrence of ‘*vitamin*’ in this sentence as <<“**■** *represents the concept vitamin.*”>>. However, the

problem is that ‘vitamin’ in natural language is ambiguous. In some cases the different vitamin types, such as vitamin A and vitamin B, are seen as instances, and in other cases as subclasses of the concept ‘vitamin’. In the first case, ‘vitamin’ is interpreted as a metaclass (a class of classes), because its instances are also classes. In the second case it is interpreted as a superclass. Therefore, the definition of ‘vitamin’ should be disambiguated by specifying what you mean: <<"■ represents the concept vitamin of which entities like vitamin A are instances.">> or <<"■ represents the concept vitamin of which entities like vitamin A are subclasses.">>. The guideline thus reads:

If a natural language fragment can both be interpreted as a metaclass or a superclass of a certain type of entities, disambiguate your definition by making a choice between one of these interpretations.

■

III.7.6 Conclusion

The research questions highlighted in the introduction of this chapter (section III.7.1 (p. 225)) and the corresponding research results achieved in this chapter and the experiment in chapter IV.3 (p. 323), are as follows.

III.7.6.1 Fostering hucolligence with Orcobas

“

Research question 2.2.1 (incentives SWiFT-game). *Which incentives for participating and improving formal-fluency does SWiFT provide, and how could the game be improved to make these incentives stronger?*

”

(Quoted from page 221.)

Conclusion 1 of research question 2.2.1. *Results related to motivation that were observed in the experiment with SWiFT-Fixtal- α include the following (see section IV.3.2.4 (p. 330)). The collaborative aspect was often mentioned as an important motivator. There was not a single player who discontinued the game for other reasons than circumstances beyond his or her control, in spite of the long duration of the session. The long duration of the game, nevertheless, was often mentioned as a counter-incentive,*

and a reason for some people rejecting the invitation to play the game. A few players expressed that their curiosity about the end-product and the intellectual challenge were motivating. Two players wanted to continue playing even after the experimenter suggested to suspend the session. Finally, players suggested improvements to the composition-record.

These results will also translate to SWiFT-Full games, because these are even more complex. This will most likely be a big hurdle for many to become and remain players.

To what extent these results are generalisable within the class of Game-Based-Orcoba has yet to be shown.

Ways to improve the incentives could include:

1. Enhance the sensation of collaboration. An example is allowing players to express discontent or satisfaction about the time another player waits before judging their node definitions, by letting them control an avatar depicting them getting angrier or happier.
2. Drastically reduce the time needed to play the game. This could, for example, be achieved by offering shorter text-fragments while working with bigger teams. The current fragments contained about 180 words, and it took about 225 minutes to translate them. This provides data for making a rough estimate of a good new fragment size. If each fragment were reduced to about 20 words (roughly a sentence) per player, the time needed would be around 25 minutes for a team in their current stage of development. Choosing a smaller total text of 200 words, would require teams of 10 persons each. Although one has to find more players, it may be much easier to do so because they do not have to invest so much time. An additional advantage of bigger teams is that the aspect of collaboration will be stressed even more. Collaboration was mentioned as an important motivator.

These results also translate to SWiFT-Full games, because these are even more complex. This will most likely be a big hurdle for many to become and remain a player. SWiFT-Fixtal, and even more so SWiFT-Full, are quite human resource-intensive with respect to the required number of fluency-players and their versatility. The player population has to be versatile in the following two senses. First, each single player must fulfill different roles with different skills, which results in a long time to learn to play the game. Second, SWiFT-Full requires different players fulfilling different roles. Therefore, the game can be played only by a team and not individually. This results in associated waiting times for (1) gathering sufficient players to form a team and (2) each other's contributions. ■

Conclusion 3 of research question 1.1.1. Aspects of the results of conclusion 1 of research question 2.2.1 on the preceding page probably also apply to the more general class of Game-Based-Orcobas. ■

The result concerning the long duration of the game inspired the following research question.

Research question 2.2.6. *[Short games] How to design a SWiFT-game in which playing a game does not take much time for the player? (That means, playing a game session takes a few seconds to at most a few minutes, and learning to play the game not more than five to ten minutes.)*

[Conclusion(s): page 299] ■

III.7.6.2 Fostering formal-fluency on a meta-level

“

Research question 2.2.2 (easier comparison translations). *How to design a SWiFT-game or variant thereof in which the quality of the translation of the players is easier to compare than in SWiFT-Full?*

(Quoted from page 221.)

”

Conclusion 1 of research question 2.2.2. *The results of adversaries are easier to compare in SWiFT-Fixtal, because the formal-target-language is fixed (see section III.7.1 (p. 225)).* ■

“

Research question 2.2.3 (no immediate full-blown formal-fluency needed). *How to design a SWiFT-game in which the player is not immediately confronted with acquiring full-blown formal-fluency, but which still has many to most of the desired effects of SWiFT-Full?*

(Quoted from page 222.)

”

Conclusion 1 of research question 2.2.3. *In SWiFT-Fixtal the structural-semantic-conditions of the formal-target-language are fixed. In the scope of SWiFT-Fixtal, this set and the notion ‘formal-target-language’ are used interchangeably. By choosing a less expressive formal-target-language, one can limit the difficulty level for the players (section III.7.1 (p. 225)). In the particular case of SWiFT-Fixtal- α the formal-target-language was SWiFT-Turtle, which is indeed a formal-language with limited expressivity. It indeed turned out that players in the game were able to create fairly high quality repres-*

entations in terms of what could be deduced with automated deductive reasoners from these representations. See section IV.3.2.1 (p. 326). ■

Conclusion 1 of research question 2.2.5. *The scoring procedure actually exposed translation weaknesses. There were several violations of the composition-record that actually reduced the score during this session (see section IV.3.2.3 (p. 327)). These would therefore also have been detected by the players themselves. An example is the meta-class/superclass ambiguity as mentioned in article 3 (p. 233). Probably, more playing hours with more text, would have exposed the other weaknesses in terms of lower scores as well to the players, inspiring them to improve their composition-record.* ■

III.7.6.3 Fostering formal-fluency on an object-level

“

Research question 2.3 (fostering formal-fluency on an object-level).
fostering formal-fluency on an object-level

(Quoted from page 143.)

”

Conclusion 1 of research question 2.3. *The experiment with SWiFT-Fixtal- α demonstrably resulted in insights of why people create sub-optimal translations, or translations that violate rules that arguably contribute to a higher quality of translations (see section IV.3.2.3 (p. 327)). These include deceiving natural language subjects, the lack of awareness of the meta-class/superclass ambiguity, a general lack of awareness of ambiguity, and reference by property. This inspired several improvements of the initial composition-record used in the experiment, as described more elaboratively in section III.7.5 (p. 239).* ■

Potential future work regarding SWiFT will be presented in chapter III.9 (p. 303).

Chapter III.8

SWiFT-Focused

(Before starting this chapter, I would like to gently remind the reader of the call for feedback on page i.)

III.8.1 Introduction and research questions

The SWiFT-Full game, but also the more restricted SWiFT-Fixtal design has a number of disadvantages:

- (1) As has been stated in research question 2.2.3 (p. 222), SWiFT-Full confronts the players immediately with full-blown formal fluency. This can be intimidating, and therefore discouraging. SWiFT-Fixtal improves this situation by fixing the formal target language: see conclusion 1 of research question 2.2.3 (p. 242). However, a further simplification turned out to be desirable.
- (2) Related to research question 2.2.4 (p. 222): SWiFT-Full is quite human resource-intensive with respect to the required number of fluency players and their versatility, as has been explained in conclusion 1 of research question 2.2.1 (p. 240). SWiFT-Fixtal is somewhat less demanding. However, for the greatest part, SWiFT-Fixtal is as resource intensive as SWiFT-Full. As already formulated in conclusion 1 of research question 2.2.1 (p. 240) the player population has to be versatile in the following two senses. First, each single player must fulfill different roles with different skills, which results in long times required to learn how to play the game. Second, SWiFT-Full requires different players fulfilling different roles. Therefore, the game can only be played by a team and not by an individual. This results in associated waiting times for (1)

gathering sufficient players to form a team and (2) each other's contributions. SWiFT-Fixtal simplifies the roles – among other things, because of the fixed formal-target-language the required reasoners do not have to be installed by the players, but can instead be built into the game by the game-developers.

- (3) As has been stated in research question 2.2.6 (p. 242), SWiFT-Full game sessions typically take a long time to play, which will most likely be a big hurdle for many to become and remain a player. This came to my awareness during experimentation with SWiFT-Fixtal. SWiFT-Full game sessions take at least the same amount of time, if not more.

III.8.1.1 Research question refinement

During the research and development process of SWiFT-Focused an additional research question arose.

Research question 2.2.7. [*Lessons general reasoner construction*] *What are lessons beyond the scope of SWiFT-Focused that can be learnt regarding the effective design of automated-deductive-reasoners?*

[*Conclusion(s): page 300*] ■

III.8.1.2 Introduction solutions

higher-order-SWiFT

The overall solution to the research questions referred to in section III.8.1 on the preceding page is *higher-order-SWiFT*, an extension of *higher-order-Orcobas*. *Higher-order-SWiFT* is a variant of the *SWiFT-class-of-games*. As described in section II.4.6 (p. 61), in *higher-order-Orcobas*, the full challenge of mastering a capability (in this case, *formal-fluency*) is broken up into smaller subcapabilities. *Higher-order-SWiFT*, therefore, already provides an answer to research question 2.2.3 (p. 222). Additionally, this break-up has a very beneficial indirect effect, which also allows a solution to research question 2.2.4 (p. 222) and research question 2.2.6 (p. 242), as will be explained later.

SWiFT-Focused

In *higher-order-SWiFT* each subcapability is addressed in a short subgame: a *SWiFT-Focused* game. Each *SWiFT-Focused* game is a direct instance of the *SWiFT-class-of-games*, i.e. it has the archetypical *SWiFT* design: a player being offered a text to translate, after which the translation is tested by comparing what can be deduced from the translation with reasoners with what logically follows from the original natural language text. The *SWiFT-Focused* games are presented to the player in order of a gradually increasing level of difficulty. It is important that the reader notes that, therefore, *SWiFT-Focused* games are an integrated part of *higher-order-SWiFT*.

Statements in this work about higher-order-SWiFT, may also implicitly relate to SWiFT-Focused (and vice versa).

All SWiFT-Focused games share the same structure, however, with another subchallenge ‘plugged’ in. Compare this with the game Pacman, in which each level is different, yet similar. A particular level in Pacman is, in a certain respect, analogous to a particular SWiFT-Focused game. Pacman as a whole, combining several levels, is analogous to higher-order-SWiFT. Note that higher-order-SWiFT is, therefore, not an instance of the SWiFT-class-of-games, but a closely related system: it combines several instances of the SWiFT-class-of-games into an, indeed, ‘higher order’ SWiFT.

The break-up of the formal-fluency capability into small subcapabilities has an additional, and essential, effect – it allows a severe reduction of the human resources needed to realise a game session. This provides a result related to research question 2.2.4 (p. 222). In SWiFT-Full it is impossible to let the computer check the quality of the translation fully automatically. For this, it should, among other things, be able to translate any natural language text flawlessly into a formal-language, which is related to the well-known A.I. hard problem of machine-translation of natural language. A.I. hard problems are problems in A.I. that are believed to require a human level of intelligence – which, to date has not been achieved, if it ever will be. However, in higher-order-SWiFT this translation *can* be made with a suitably chosen subcapability. The subcapability has to be chosen such that (1) it can be adequately tested with a predefined formal-language and corresponding, very limited, subset of natural language texts and (2) the translation between the latter can be done automatically. Often, sufficiently limiting the problem domain, allows it to come within the reach of automated solving. The limited domain, albeit not the original full problem, however, can still be of much use. Compare this with the fact that speech recognition is A.I. hard as well, but works well if you restrict the domain, allowing some useful applications in automation of call-centers. An example is a restriction of the domain to train ticket reservations. A similar approach seems to be effective in the case of fostering formal-fluency as shown in the rest of this chapter. Additionally, in higher-order-SWiFT the following ‘trick’ is applied to make the translation even more reliable. Instead of generating or selecting a natural language text first, the computer starts with creating a representation in the formal-language. Then it translates this representation ‘under the hood’ into natural language. Translation from formal-language to natural language is far easier than translation in the opposite direction. Hence, when the player is presented the natural language text, the computer already ‘knows’ its formal-language representation. All this allows higher-order-SWiFT to be designed as a single player game, in which the player only fulfills the role of translator from natural language. All other roles are fulfilled by the computer: creation of the natural language text; the questions for the question-attack-round and the deduction-algorithms for the algorithmic-defence-round. This means that a player can start

right away playing the game, and does not have to wait for the formation of a team (providing an answer to research question 2.2.4 (p. 222)).

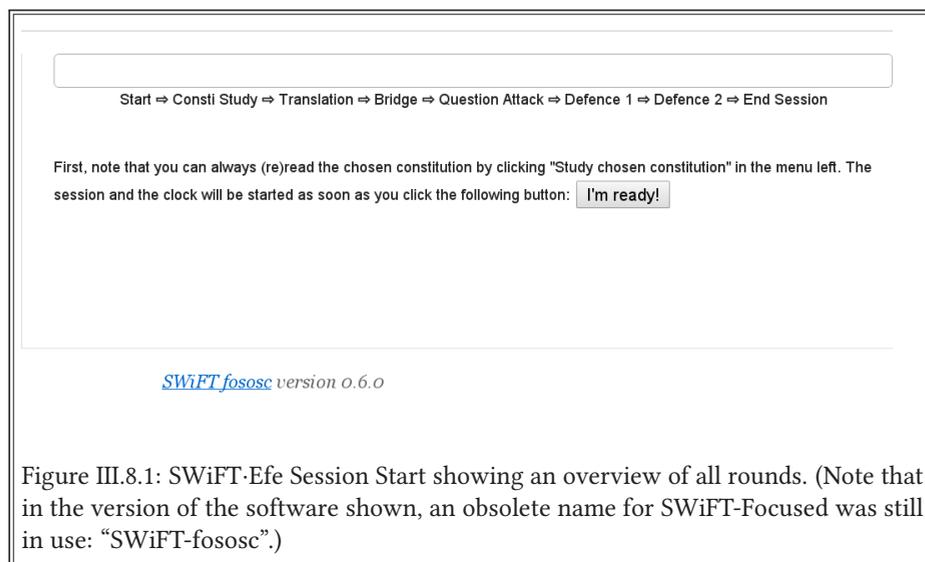
Yet another positive effect of the break-up is that it allows each SWiFT-Focused game to be designed such that each translation session is on average playable in a timespan of a few seconds to at most a few minutes, and that, on average, learning how to play the game does not occupy more than five to ten minutes (providing an answer to research question 2.2.6 (p. 242)).

It is important to note that higher-order-SWiFT is *not* intended as a replacement for SWiFT-Full. The latter has, in its turn, some major advantages over higher-order-SWiFT. The SWiFT-Focused games proceed in the following rounds. (Used is a running example of the SWiFT-Efe game, a SWiFT-Focused game developed in this chapter. Moreover, an overview of all rounds are displayed on the first page that is shown to the player, as can be seen in Fig. III.8.1.) *Consti-study-round*: a composition-record is shown which explains how to create a translation in the formal-target-language. *Translation-round*: a sentence or text in English is given (the source text). The player should translate it as swiftly and correctly as (s)he can into the prescribed formal-language. *Bridge-construction-round*: the computer asks for additional information to establish how to connect the words in the player's translation to concepts that are expressed in the English text. It needs this in the background for the rounds to come. *Question-attack-round*: the computer will now attack the player's translation by posing (natural language) questions that have a very specific answer based on the English source text. The game then presents the answer as it should be based on the English source text, and asks the player whether it is the only correct one. *Algorithmic-defence-round*: the computer now constructs a deduction-algorithm, purely based on information in the question, which is able to deduce the answer to the question according to the player's translation.

*consti-study-
round*

III.8.1.3 Implementation of SWiFT-Focused

A considerable part of SWiFT-Focused, including several SWiFT-Focused instances have been implemented by the author in the role of lead developer and several contributors. It is the most comprehensive implementation of created in the scope of SWiFT R&D to date. The source-code repository can be accessed at <https://github.com/swiftgame/SWiFTfososc>. The examples throughout this chapter have been taken from running instances of this implementation.



III.8.2 SWiFT-Focused rounds in detail

This section presents a detailed description of the SWiFT-Focused rounds and uses SWiFT-Focused with the Efe challenge (SWiFT-Efe) as a running example. (For the purposes of this section it is not necessary to understand all details of the latter challenge. It will be explained in detail in section III.8.6 (p. 288).)

III.8.2.1 Consti·study·round

First, to determine the total time the player spends studying the composition-record, the player must hold down a key while reading. The composition-record is hidden by default (see fig. III.8.2 on the next page), and only shown when a specific key is held down (see fig. III.8.3 (p. 251)). The study timer exclusively runs in the background when the key is pressed. The reason for this is that it is now more likely that the timer runs when the player is actually reading – most people will probably let go of the keyboard when they are distracted, for example when they have to go to the toilet or want to get a drink. Thus, this allows for a more precise measurement of the study time.

As soon as the player presses the required key(s), the studying can begin, as shown in fig. III.8.3 (p. 251).

Study Constitution

Start ⇒ **Consti Study** ⇒ Translation ⇒ Bridge ⇒ Question Attack ⇒ Defence 1 ⇒ Defence 2 ⇒ End Session

Study the constitution below by holding the 'g' key down on your keyboard.

Info

Creationdate:	09-06-2014 23:24
Created by:	Admin (id: 1)
Version:	09-06-2014 23:24

If you want to read on, please hold the 'g' key.

Figure III.8.2: Consti-study-round of SWiFT-Efe with hidden composition-record.

III.8.2.2 Translation-round

*translation-
round*

In the *translation-round*, the player has to type in the best translation (s)he can produce as quickly as possible. A translation that is grammatically incorrect, is not accepted for submission (see fig. III.8.4 (p. 252)), otherwise it is (see fig. III.8.5 (p. 253)).

III.8.2.3 Bridge-construction-round

*bridge-
construction-
round*

In SWiFT-Efe the 'bridge' between natural language and the formal-language is established by expressing the relation between the constants of the formal-target-language and the nouns in natural language. This process takes place in the *bridge-construction-round*. In SWiFT-Efe the nouns are proper names. As can be seen in fig. III.8.6 (p. 254), the interface makes this task easy, by already listing all nouns in the source text, and providing the constants in a drop-down box.

III.8.2.4 Queries as algorithms

In the traditional view, a query is merely a specification of a question, and a reasoner is the algorithm that answers that question. In SWiFT, however, queries for reasoners are also considered to be computer programs, or algorithms, in a very high program-

Constitution EfeLang

Challenge: efe

1 Definition of the target language: efeLang

The target language efeLang is defined as follows. A document in efeLang consists of a sequence of sentences, and sentences of a sequence of words and auxiliary symbols. Lets first explain the words, then the sentences and after that the documents. Note that no other language construct than as defined exactly in this document is allowed.

1.1 Words

The words in efeLang come in two flavours: constants and (the B and F) predicates.

1.1.1 Constants

A constant is a word that stands for a certain entity (which could be a person, a table, an armadillo, you name it). Examples of constants in efeLang are:

- **c**
- **c1**
- **obama**

Figure III.8.3: Consti-study-round of SWiFT-Efe with visible composition-record.

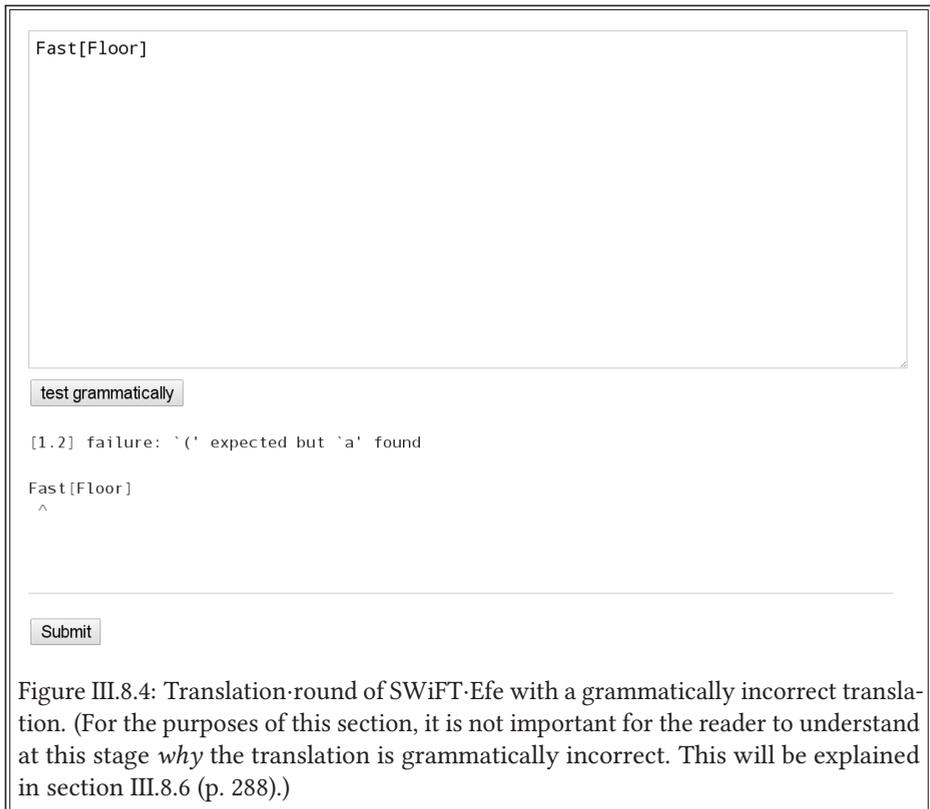


Figure III.8.4: Translation-round of SWiFT-Efe with a grammatically incorrect translation. (For the purposes of this section, it is not important for the reader to understand at this stage *why* the translation is grammatically incorrect. This will be explained in section III.8.6 (p. 288).)

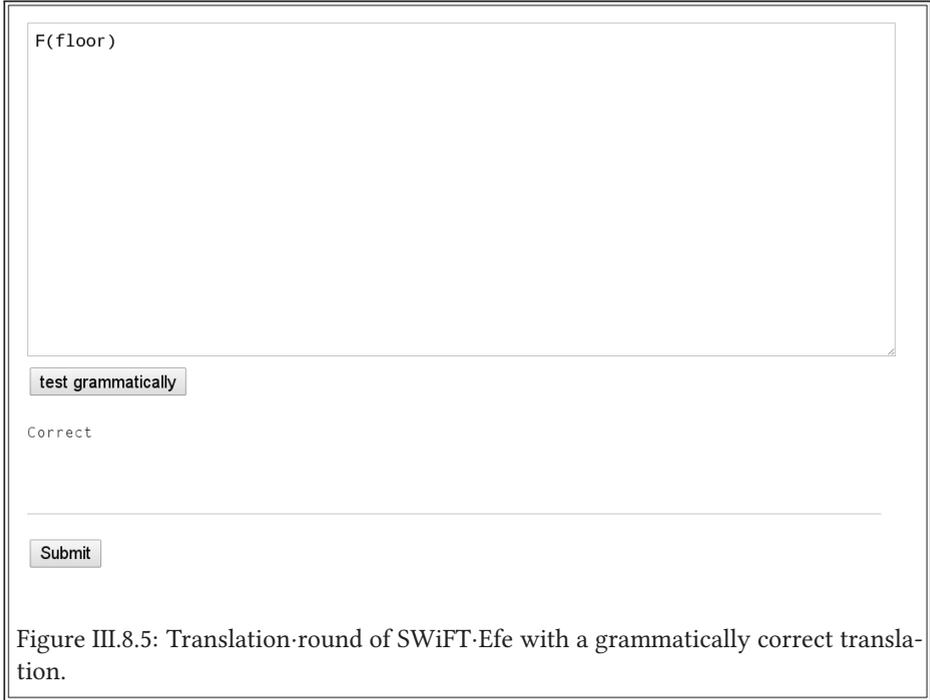


Figure III.8.5: Translation-round of SWiFT-Efe with a grammatically correct translation.

Bridge Construction Round

Start ⇒ [Consti Study](#) ⇒ [Translation](#) ⇒ **Bridge** ⇒ Question Attack ⇒ Defence 1 ⇒ Defence 2 ⇒ End Session

Source

Floor is fast.

Translation

F(floor)

To which person, being or thing in natural language does the constant **floor** refer?

Please select one of:

Figure III.8.6: Bridge-construction-round of SWIFT-Efe.



ming language. (This has been explained in more detail in section III.6.4.2 (p. 219).) The reason is that this allows a comparison to be made between *any* expression in *any* computer-language intended to specify a deduction. In other words, it unifies all these languages under one common notion. This unification is needed in some variants of SWiFT. This is not the case for SWiFT-Focused. However, also this section adopts this unified terminology, among other things to use terminology consistently throughout this book.

III.8.2.5 Question-attack-round

In the question-attack-round, the computer generates one or more (natural language) questions based on the source text and the translation created by the player (example in fig. III.8.7). Note that under the hood, the computer first generates a formal representation of the question (in the question-formal-language), and then translates it into natural language. Also, note that the question in the formal-language, in the running example, is equal to the deduction-algorithm, using the unified terminology of ‘queries as algorithms’ (section III.8.2.4 (p. 250)). Therefore, the following round, the algorithmic-defence-round, is in fact already prepared. This equivalence will hold more generally, so, for most SWiFT-Focused games. In the particular example of fig. III.8.7, there is only one question created.

The answer that the computer assumes to be correct is first offered to the user for verification. (This is not implemented in the running example.) The reason for this is that even though the deduction from the formal representation produced by the

computer is correct, the representation itself may be an incorrect translation of the natural language text. It is impossible to foresee all peculiarities of natural language semantics: this is a property inherent to natural language. Feedback of the users is used by the SWiFT-Focused developers to fine-tune the computer's translation process in future versions of the game.

III.8.2.6 Algorithmic-defence-round

The algorithmic-defence-round consists of two stages: in the first stage the algorithm is constructed, in the second it is executed. Let the reader be reminded I employ the unified 'query as algorithms' terminology (section III.8.2.4 (p. 250)).

*algorithmic-
construction-
stage*

In the *algorithmic-construction-stage* deduction-algorithms are constructed that deduce the answers to the questions (from the question-attack-round) from the translation of the player. These deduction-algorithms are also generated by the computer. An example is shown in fig. III.8.8 on the facing page. Note that it shows only one algorithm, because there was only one question.

The deduction-algorithms are created by the *computer*, and not by players, for three reasons. First, creating deduction-algorithms is not (always) the easiest of tasks. SWiFT-Focused does not bother the player with this: it lets the player purely focus on the translation process. Second, because it will be hard, if not impossible for the computer to verify if the deduction-algorithms constructed by the player are legitimate. In SWiFT-Full the verification is done by the human adversaries. SWiFT-Focused, however, is a single player game. Third, because the computer already has created these deduction-algorithms, or is close to it because it has already created formal-language questions during the question-attack-round (see section III.8.2.5 on the previous page) – and these are close, if not identical to the deduction-algorithms.

*algorithmic-
execution-stage*

In the *algorithmic-execution-stage* the constructed deduction-algorithms are executed against the translation of the player. Figure III.8.9 (p. 258) shows the result of executing a deduction-algorithm. It also shows the correct answer. If the same answer was deduced from the translation of the player, this contributes positively to his or her score. In the case of the running example, there was only one question. It was answered correctly. Therefore, the correctness score is here 100%.

The coming sections present a number of SWiFT-Focused instances following the template provided in IV.3.1. These are SWiFT-Efe and SWiFT-NoUna. However, for clarity, the following section, section III.8.5 (p. 264), first presents and defines the formal-languages used by these instances.

Algorithmic Defence Round

Start ⇒ [Consti Study](#) ⇒ [Translation](#) ⇒ [Bridge](#) ⇒ [Question Attack](#) ⇒ **Defence 1** ⇒ Defence 2 ⇒ End Session

Recap...

Question in English

Mention people or things which are fast. And... do not mention some, but mention all of them!

Our answer based source text

Floor is fast.

Defence Algorithm

We present you the following, just to give you a brief inside view in the kitchen of the cook. It doesn't matter if you don't understand it, it is just to show you that it is cooked professionally.

```
MostInfo(PatVar(s),Forall(Var(name = x),PatVar(s),PredApp_Plofofa(Predicate(name = F, arity = 1),List(Var(name = x))))))
```

Start Derivation!

Figure III.8.8: Algorithmic-construction-stage of SWiFT·Efe. (For the purposes of this section it is not necessary to understand the defence algorithm. It will be explained in detail in section III.8.5 (p. 264) and section III.8.6 (p. 288).)

Recap...

Question in English

Mention people or things which are fast. And... do not mention some, but mention all of them!

Algorithmic Defence in CTL

MostInfo(PatVar(s),Forall(Var(name = x),PatVar(s),PredApp_Plofofa(Predicate(name = F, arity = 1),List(Var(name = x))))))

Results

	Derived from source	Derived from your translation
in CTL	Forall(Var(name = x),List(Constant(name = ctNameFloor)),PredApp_Fofa(Predicate(name = F, arity = 1),List(Var(name = x))))	Forall(Var(name = x),List(Constant(name = floor),PredApp_Fofa(Predicate(name = F, arity = 1),List(Var(name = x))))
in English	Floor is fast.	Floor is fast.

The answer derived from your translation is completely correct. You won!

Figure III.8.9: Algorithmic execution stage in SWIFT-Efe.

III.8.3 Performance metric options

Development of a suitable performance metric expressing the degree of formal-fluency of a player, is not trivial. This gives rise to an additional research question:

Research question 2.2.4. *[Performance metric] What is a good performance metric for the performance of the players of the SWiFT-Focused games?*
[Conclusion(s): page 300] ■

Solutions to this question are presented below. However, the following first provides some notations and terminology.

Definition 45 (Performance Metric Notations).

- *translation-duration* (trans_dur): the duration of the translation process of the player of a given text. (This includes the time (s)he spends on studying the composition-record during the translation process.)
- S_ques : the set of all questions produced in the question-attack-round. (In the example in section III.8.2 (p. 249), SWiFT-Efe, there is only one question produced per translation round.)
- $\text{source_size}_{\text{symbols}}$: the size of the source text that is translated in the number of symbols it contains.
- trans_speed : standard notation for translation speed. In different metrics, it may be defined differently.
- $N_ques = |S_ques|$: the number of questions produced in the question-attack-round.
- ques_i : an indexed question from the question-attack-round. (So, actually this is a function of type $\{1, \dots, N_ques\} \rightarrow S_ques$.)
- $\text{cor_ans} : S_ques \rightarrow [0, 1]$: the correctness of the answers to questions. The correctness can be a real number from from 0 (completely incorrect) to 1 (completely correct). (In SWiFT-Efe there are only two options: 0 or 1.)
- score_constant : some constant with which the core of the metric is multiplied to get numbers that are more ‘pleasant’ to read. Without this constant, scores such as ‘0.00138’ could emerge. Note that it does not have any semantic effects: It does not affect the ratios between the scores of players. ■

The first candidate for the metric is defined as follows. (An explanation follows after it.)

Definition 46 (`basic_performance_level_metric`).

$$\text{score_constant} \cdot \frac{\sum_{q \in S_ques} \text{cor_ans}(q)}{|S_ques|} \cdot \text{trans_speed},$$

where

$$\text{trans_speed} = \frac{\text{source_size}_{\text{symbols}}}{\text{trans_dur}}$$

■

The definition of `basic_performance_level_metric` is sufficiently self-evident, therefore I omit an explanation. One problem with it, is that the speed of the translation process (`trans_speed`) is based on the number of symbols in the source being translated per unit of time (`source_sizesymbols`). The latter may not be the best way to measure the size of the translation for this purpose. Let me first parameterise the previous definition to allow alternative definitions for the size of the source to be ‘plugged in’:

Definition 47 (`basic_performance_level_metric_class[source_size]`). *The class of performance level metrics `basic_performance_level_metric_class[source_size]` is obtained by taking `basic_performance_level_metric` (definition 46) and turning `source_sizewords` into a parameter `source_size`, with a yet unspecified submetric for source-size. In other words, its definition is as follows:*

$$\text{score_constant} \cdot \frac{\sum_{q \in S_ques} \text{cor_ans}(q)}{|S_ques|} \cdot \text{trans_speed},$$

where

$$\text{trans_speed} = \frac{\text{source_size}}{\text{trans_dur}}$$

■

Note that `basic_performance_level_metric` now is a special instance of this class, to wit `basic_performance_level_metric_class[source_size = source_sizesymbols]`.

A problem, however, is that two language expressions with different symbol-sizes may still be considered to be of the same length from the perspective of the translation process. For example, two expressions have, translation-wise, an identical size if they are the same except for the names of persons. For example, compare “Akwası is a good fellow.” and “Jim is a good fellow.” These are identical translation challenges, while their symbolic lengths differ. An alternative measure is based on the number of *words* being translated, so:

Definition 48.

$source_size_{words}$: the size of the source text is translated in the number of words it contains. ■

$basic_performance_level_metric_class[source_size = source_size_{words}]$ solves the latter problem. However, it may still not be best measure. This can also be made clear with an example of a translation. The text “Akwasi, Veerle, John-Jules, [here 990 more names], and Rapunzel still believe in Santa Claus.”, contains, translation-wise, much less information to be translated than a 1000 word summary of Plato’s *Parmenides*, while both texts contain the same number of words. I suspect that a metric can often (if not always) be improved by investigating the semantics of the (type of) natural language fragments that are being translated. This implies that it would be impossible to define the best metric in advance, it has to be improved ‘along the way’. Nevertheless, one could use one default metric, that performs reasonably well, as long as a better one is not (yet) available.

Moreover, note that providing a sufficiently large sample of heterogeneous texts to translate, will average out differences, and may produce a reliable score for fluency using the latter metric. However, the disadvantage, is indeed, the large(r) sample size needed, requiring more human resources (in this case: playing time).

III.8.3.1 Future work

One problem with the scoring procedure is that it does not keep into account the total time the player spent studying the composition-record. However, I believe it is reasonable to assign a higher score to a player P if P ’s studying time is lower, when everything else identical. A metric for this purpose is left as future work, because it is not trivial to integrate it in the performance metric in a satisfactory way. Moreover, it is not an urgent matter: in higher-order-SWiFT there is a maximum composition-record study time that is considered to be reasonable (thus, not too short, and not too long). This, at the least, means that the score of P was achieved under the condition that the study time was of a reasonable length. This is not ideal, yet, of course, because it does not allow one to differentiate players who spent less study time.

Moreover, a further refinement of integrating the composition-record time is to differentiate also between the time spent on studying the composition-record *before* and *during* the translation process. (The metrics presented in section III.8.3 (p. 259) counts the latter simply as translation time.)

III.8.4 SWiFT-Focused design process and template

SWiFT-Focused requires the development of a great many instances to cover a considerable number of aspects of formal-fluency.

Each specific SWiFT-Focused instance, so one that focuses on a specific subchallenge in fostering formal-fluency, requires the development of a number of ingredients. This development process is far from trivial, and requires quite some creativity from the designer, amongst other things because one needs to design or select suitable formal-languages, text-challenge-generators, question-generators, parsers and reasoners. Most of these components may not yet exist, or not be readily available, because they are highly dependent on the focus. They have to be ‘custom-made’ for the focus. For example, a specific focus may often require a specific, not yet existing, formal-language to be designed, to isolate the challenge, and not overburdening the player with languages that are unnecessarily rich. Specific questions may require specific reasoning problems to be solved that are not readily solvable with the existing repertoire of reasoners.

The previous is exemplified by the design process of one of the first SWiFT-Focused instances. I expected that implementing a SWiFT-Focused game would ‘only’ require defining translations between the designed ingredients and an existing reasoner and its associated formal-language. However, it turned out that this implementation can be much more involved. This can even be the case with a seemingly ‘harmless’ and simple looking SWiFT-Focused instance. For example, the design of the SWiFT-NoUna-game includes a type of questions that could not be answered by any of-the-shelf reasoner known to me. It required the combination of two existing reasoners: *Eprover* (Schulz, 2013) and *Paradox* (Claessen and Sörensson, 2003). Both the specification of the reasoning algorithm and the implementation by having to stitch together two reasoners took considerably more time than expected at the outset.

Eprover
Paradox

This gives rise to an additional research question:

Research question 2.2.8. *[Accelerate SWiFT-Focused instance design] SWiFT-Focused requires the development of a great many instances to be developed to cover most aspects of formal-fluency. How to design an approach in which this development can take place in an accelerated fashion?*

[Conclusion(s): page 301] ■

As a first step toward providing an answer to this question, I believe it is important to at least streamline the design process, in the following done by providing a *design template*, intended to be used by SWiFT-Focused-instance developers. The template

consists of the following elements, each describing an ingredient of the SWiFT-Focused-instance.

- Challenge in Focus: A short description of the particular challenge that is being focused on.
- Challenge-set: Definition of the challenge-set, which is a set of natural language texts.
- Question Set: Definition of the type of questions that are going to be asked. In SWiFT-Full any question can be posed as long as it can be answered based on the given natural language text. However, in SWiFT-Focused, there is often a certain problem that is in focus, which requires posing certain questions, while other questions would be irrelevant given the focus.
- Formal-languages. Definition of the following formal-languages:
 - The *formal-target-language*: The player must translate the given natural language text into the formal-target-language. *formal-target-language*
 - The *text-generation-language*: The text-generation-language is an extension of the formal-target-language (not necessarily a proper extension, so they may be equal to each other). It is needed for the piece of software that automatically generates a natural language text that is to be offered to the player. The formal-target-language itself is not necessarily sufficiently expressive in SWiFT-Focused games, because there may be a difference between the representation required to generate a good natural language text, and the semantic conditions in focus. (An example is the text-generation-language of SWiFT-NoUna, which will be introduced later.) *text-generation-language*
 - The *question-formal-language*: The formal-language in which the goal for the reasoner has to be defined. *question-formal-language*
 - The *answer-language*: which is complementary to the question-formal-language. *answer-language*
 - The *bridge-language*: The bridge-language is the (grammar and syntax) dictionary between the given formal-languages and natural language. *bridge-language*

The naming convention used for a specific SWiFT-Focused instance, so an instantiation of the template above, is SWiFT-[label of the challenge]. A formal specification of the SWiFT-Focused instances developed by me, following this design template, to wit SWiFT-Efe and SWiFT-NoUna are given in section III.8.6 (p. 288) and section III.8.7 (p. 292).

III.8.5 The formal languages

After providing some general definitions and terminology, this section specifies the formal languages that are used by one or more of the defined SWIFT-Focused instances, and languages that help define these.

III.8.5.1 General definitions and terminology

III.8.5.1.1 Pattern languages

Reasoners for formal languages are mostly goal-oriented: can a specific sentence, or set of sentences be deduced from a given KR-base in a given formal language? In this way, a reasoner becomes a useful tool for users because it allows them to deduce answers to their questions. Therefore, such reasoners require the specification of a *KR-language* and a language to specify reasoning goals: the query language. (A *KR-language* is a formal language that is used as a language to represent knowledge about which (an) automated deductive reasoner(s) reason(s).) Query languages are often designed as an ad-hoc mixture of ‘formalised’ natural language questions, and a collection of ad-hoc language constructs. I will now present an approach that, I believe, is more elegant and practical, by unifying querying with knowledge representation.

KR-language

The approach is based on defining a sufficiently expressive formal language that allows one to express the structure that the answer should meet. In the approach, the design of a query language is as follows. First, the *KR-language* of the reasoner is extended to another formal language, in this context called the answer language. The reasoner will provide its reasoning result, hence its *answer*, in the answer language. The answer language in the sense just given is crucial to reach the desired expressivity. Second, a language is defined that enables specifying the structure the answer (in the answer language) should meet. This language is used as the query language. A structural and elegant way to do this, is by taking the answer language as the basis for a pattern language.

Note that the extension of the *KR-language* is often necessary. In general, the formal language of the *KR-base* will be too restrictive to serve as the answer language for the intended usage. For example, if one would like to deduce all constants for which $\varphi(x)$ holds from a *KR-base* in first-order logic, it would be tedious, if not impossible, to state the reasoning goal in FOL itself: one would be obliged to sum up all sentences of the form $\varphi(x)$, where x is substituted by a constant from the (signature of the) given FOL. If it is a *KR-base* with a few constants, it is inconvenient, with one with a million it would be virtually impossible.

On the other hand, one could now be tempted to extend the *KR-language* to the

answer-language – why not? The reason is that keeping them distinct enables the reasoner designer to control reasoning complexity. The answer-language may lead to deductions that the designer considers unacceptably complex if it is used to also express statements in the KR-base. Keeping them separated, provides the designer with an essential tool to strike a much better balance between the expressivity of the query-language, the KR-language and the resulting reasoning complexity.

Traditionally, in many reasoners a part of the query-language indeed consists of a pattern-language based on the KR-language, however, because of the limited expressivity of most KR-languages, one has to also incorporate ad-hoc ways to express questions. A solution to this problem is to define a specific question-formal-language, which in most cases will be an extension of the given KR-language

As a generic way to define query-languages, this book uses the notion *pattern-language*. These languages consist of a modification of a KR-language, in which certain syntactical structures are allowed to be replaced by variables, so-called *pattern variables*. The semantics of a sentence of a pattern-language (a *pattern*) is that one sentence represents a, possibly infinite, set of sentences from the KR-language: namely all sentences that can be formed by (exclusively) substituting the pattern variables.

pattern-language
pattern variable
pattern

The desired results of a reasoner can now be defined as follows. First the notions complete and sound reasoner are defined.

Definition 49 (Sound reasoner). *For any KR-base K in formal-language L_K and a pattern π in pattern-language L_P (where L_P is compatible with L_K), a sound reasoner compatible with these languages, deduces the disjunction of the transitive closure under deduction of K and the set indicated by π , or a subset of the latter, including the empty set. Another way to phrase it is that it should produce the set or a subset¹ of sentences that are deducible from K and match the pattern π . ■*

A sound reasoner produces only correct answers, but perhaps not all of them. A *complete* reasoner for these languages is a reasoner that is described by replacing the word ‘subset’ by ‘superset’ in the definition of the sound reasoner, so:

Definition 50 (Complete reasoner). *For any KR-base K in formal-language L_K and a pattern π in pattern-language L_P (where L_P is compatible with L_K), a complete reasoner compatible with these languages, deduces the (set-theoretic) disjunction of the transitive closure under deduction of K and the set indicated by π , or a superset² of the latter. Another way to phrase it is that it should produce the set or a superset of sentences that are deducible from K and match the pattern π . ■*

¹ Every set is also a subset and a superset of itself, however, for clarity ‘the set’ is explicitly mentioned.

²See footnote 1 on page 265

Definition 51 (Sound and complete reasoner). A sound and complete reasoner is both sound and complete in the senses just described. ■

The latter is a property an ideal reasoner should meet: it produces *all* of the, and *exclusively*, correct results.

Some terminology: in this book, pattern-language names will always start with the prefix pl- (Pattern Language).

III.8.5.1.2 The `mostInfo`-constructs

A problem with pattern-languages is that not all of them are always sufficiently restrictive. This is due to the fact that for each sentence that is deducible and matches the pattern, also weaker sentences may match the pattern. A sound and complete reasoner would produce all such weaker sentences as well. For example, if one asks “Who are the children of Akwasi”, one expects an answer that includes all of his children, and not only some of them. Depending on the formal-languages used, however, it may be so that the reasoner also produces sentences with strict subsets of Akwasi’s children. An example of this problem is provided in section III.8.5.5 (p. 272).

The underlying problem is that pattern-languages as defined so far do not allow one to impose an additional restriction on the desired answers: to produce the *most informative match* given the KR-base and the pattern. For this reason, this work introduces an additional language construct that can be added to an existing pattern-language: the *mostInfo-construct*. This is a meta construct that indicates which pattern variables should be substituted with the most-informative syntactical match. An example can be found in section III.8.5.5 (p. 272).

mostInfo-construct

III.8.5.2 First-order logic

First-order logic (FOL) is among the most well-known symbolic logics (Dalen, 1997)³. In spite of the fact this work assumes that the reader knows FOL, this section provides an explicit definition of its syntax. The reasons include that (1) several extensions and simplifications of FOL are presented in the following sections that are based on this syntax definition, (2) this work applies an alternative syntax because the sentences have to be easy to write on a computer. Therefore, the syntax only includes symbols that are readily available on a standard computer keyboard.

There are different varieties of FOL. In this work the word ‘FOL’ denotes FOL with implication, negation, conjunction, disjunction and equality, but without functions.

³Actually, it is a closely related family of logics, e.g. there are variants with or without function symbols, equation etc.

III.8.5.2.1 Syntax

First, to make it easy for the reader with knowledge about FOL, the following defines the syntax by describing its differences with a very common notation of FOL (Dalen, 1997):

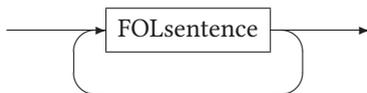
- (1) The \forall quantifier is replaced with `forall`.
- (2) The \exists quantifier is replaced with `exists`.
- (3) Implication \rightarrow is replaced with `->`.
- (4) Conjunction \wedge is replaced with `and`.
- (5) Disjunction \vee is replaced with `or`.
- (6) Variables and constants are a sequence of decimal digits and letters from the (standard ISO) Latin alphabet, and should start with a lower case letter from the latter. Examples are `c1`, `x` and `aKwasi4Dok001`.

Second, the following provides a formal definition using EBNF-syntax diagrams. However, a note is required. The (modern) notion ‘FOL’, as used in this work, is in fact not one language, but a family of languages (Dalen, 1997). Each member of this family is specified by the predicate symbols and associated arity, and the set of constants that may be used in the formulas. This is called the *signature* (Hodges, 2013) or *structure* (Dalen, 1997).⁴

*signature
structure*

The definition of the syntax of a FOL \mathcal{L} is then as follows. Let the signature be defined as follows $\{PredicateSymbol_1 : arity_1, \dots, PredicateSymbol_n : arity_n\}$, where $PredicateSymbol_i$ is a predicate symbol of \mathcal{L} with arity $arity_i$. For example, $PredicateSymbol_3 = isFatherOf$ and $arity_3 = 2$. A predicate symbol cannot be equal to `isEqualTo`, because this word is already reserved, as will become clear in the sequel.

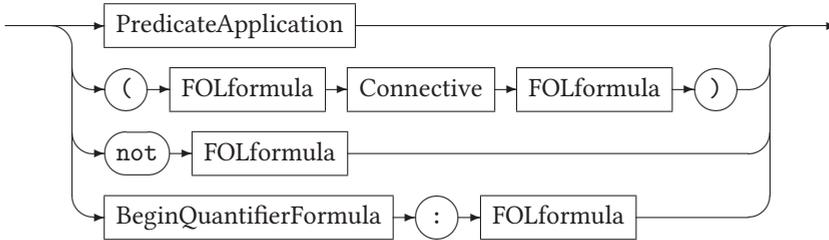
FOLdocument



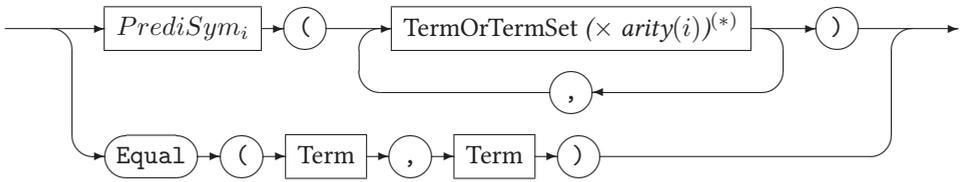
⁴Note that there is an alternative, traditional, approach in which there is only one FOL (Wikipedia, 2015) The chosen approach, however, is more suitable for when the logic is applied for knowledge representation, rather than an abstract tool for investigating symbolic logics.

Where FOLsentence is a FOLformula in which all variables, if any, are bound.⁵ FOLformula is defined in the following grammar.

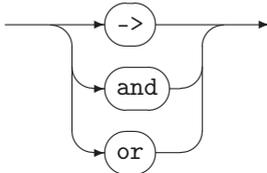
FOLformula



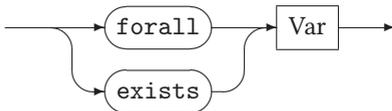
PredicateApplication



Connective



BeginQuantifierFormula

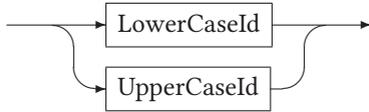


Term



⁵FOLsentence is not defined in a production rule, because the notion 'boundedness' cannot, to the best of my knowledge, be expressed elegantly using a context-free grammar. This is because the syntax used for FOL is context-sensitive (Ganea, 2013).

PrediSym_i



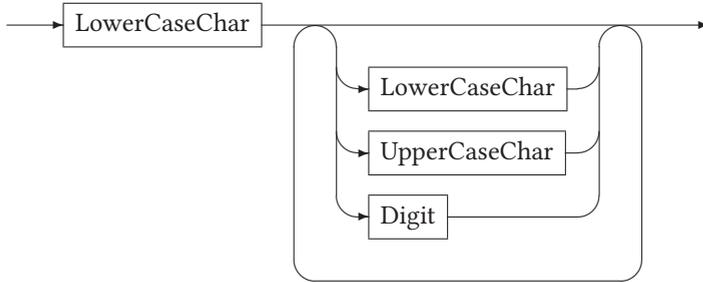
Constant



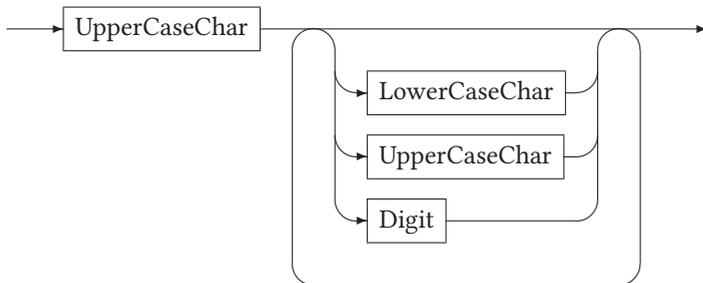
Var



LowerCaseId



UpperCaseId



where LowerCaseChar, UpperCaseChar, and Digit, respectively represent a lower case, an upper case character of the (ISO basic) Latin alphabet and a decimal digit.
 (*) repeat the syntax-loop arity(*i*) times.

III.8.5.3 Efe-language

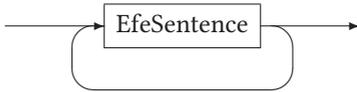
Efe-language is a minor fragment of first-order logic. It has only two unary predicates (predicates with one argument), B and F (with the meaning ‘big’ and ‘fast’). The following examples cover the complete expressivity of the language:

$B(c1)$
The B predicate applied to constant c1, i.e. the entity referred to by c1 is big.

$F(akwasi)$
The F predicate applied to constant akwasi, i.e. the entity referred to by akwasi is fast.

The syntax of Efe-language is defined as follows.

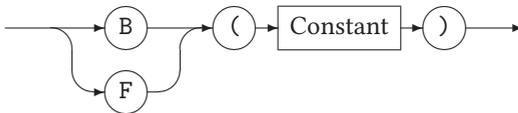
EfeDocument



EfeSentence



PredicateApplication



where all non-terminals not defined in the diagram above, are as defined in previously occurring syntax diagrams.

III.8.5.4 Folsequa-language

Folsequa-language

Folsequa-language (FOL Set Quantifier) is equal to FOL extended with quantifiers with (finite) sets of constants.

Examples of Folsequa-language sentences are the following.

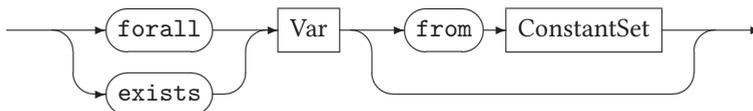
forall x from {c, c2, akwasi}:P(x)
P holds for every constant in the set {c, c2, akwasi}

exists y from {c4, akwasi, c112, d3}: forall x from {c, c2, akwasi}:P(x,y)
There exists a y from {c4, akwasi, c112, d3} such that P(x,y) holds for every x in the set {c, c2, akwasi}

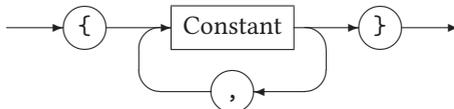
The syntax of Folsequa-language is defined by the following replacements in the syntax diagram of FOL (see section III.8.5.2.1 (p. 267)):

- (1) Replacement of *FOLdocument* with *FolsequaDocument*.
- (2) Replacement of all occurrences of the word *FOLformula* with *FolsequaFormula*.
- (3) Replacement of the *BeginQuantifierFormula* of the syntax diagram of FOL with the following production rule:

BeginQuantifierFormula



ConstantSet



The semantics of the language is defined by means of translation to FOL. The translation function T_{folsequa} from Folsequa-language sentences to FOL sentences is inductively defined as follows.

$$T_{\text{folsequa}}(\varphi) = \varphi \text{ iff } \varphi \text{ does not contain set quantifiers.} \tag{III.8.1}$$

$$T_{\text{folsequa}}(\varphi_1 \square \varphi_2) = T_{\text{folsequa}}(\varphi_1) \square T_{\text{folsequa}}(\varphi_2) \tag{III.8.2}$$

for any formula φ , where \square is any of the connectives \rightarrow , and or or.

$$T_{\text{folsequa}}(\text{not } \varphi) = \text{not } T_{\text{folsequa}}(\varphi) \tag{III.8.3}$$

for any formula φ .

$$T_{\text{folsequa}}(\text{forall } v \text{ from } \{c_1, c_2, \dots, c_n\} : \varphi(v)) = T_{\text{folsequa}}(\varphi(c_1)) \text{ and } T_{\text{folsequa}}(\varphi(c_2)) \text{ and } \dots \text{ and } T_{\text{folsequa}}(\varphi(c_n)),$$

iff v is the only variable that occurs free in $\varphi(v)$. (III.8.4)

$$T_{\text{folsequa}}(\text{exists } v \text{ from } \{c_1, c_2, \dots, c_n\} : \varphi(v)) = T_{\text{folsequa}}(\varphi(c_1)) \text{ or } T_{\text{folsequa}}(\varphi(c_2)) \text{ or } \dots \text{ or } T_{\text{folsequa}}(\varphi(c_n)),$$

iff v is the only variable that occurs free in $\varphi(v)$. (III.8.5)

III.8.5.5 Pattern-languages Plosequa-language and Plosequamoo-language

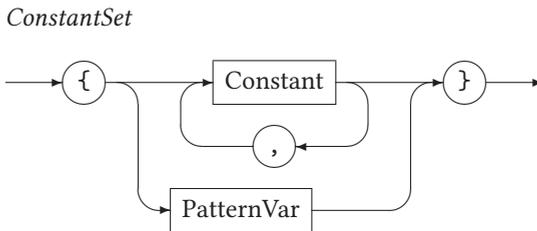
Plosequa-language

Plosequa-language is a pattern-language based on Folsequa-language. In it the set of a set quantifier may be replaced with a pattern variable. The following are examples of Plosequa-language sentences.

<code>forall x from s_ : P(c2, x)</code>
Pattern representing the set of sentences that can be obtained by substituting the pattern variable <code>s_</code> by a set of constants.

The syntax of Plosequa-language is defined by the following replacements in the syntax diagram of Folsequa-language (see section III.8.5.4 (p. 270)):

- (1) Replacement of *FOLdocument* with *PlosequaDocument*.
- (2) Replacement of all occurrences of the word *FOLformula* with *PlosequaFormula*.
- (3) Replacement of *ConstantSet* of the syntax diagram of Folsequa-language with the following production rules:



PatternVar



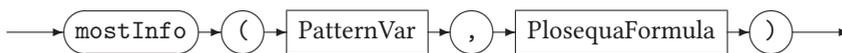
Plosequa-language, however, does not offer sufficient expressivity. If we, for example, pose the question who the children of Akwasi are, we are not asking for just any value that is correct, but the most informative value that can be determined based on the content of the KR-base. This is solved in *Plosequamo-language* (Plosequa-language with Most Informative Restriction), which is formed by extending Plosequa-language with a `mostInfo`-construct, as explained in the following.⁶

<code>mostInfo(s_, forall x from s_:P(c2, x))</code>
Select the most informative sentence(s) from all deducible sentences that match the pattern: in this case these are the sentences of the form <code>forall x from s_:P(c2, x)</code> with the <i>largest</i> set <code>s_</code> .

In Plosequamo-language only one pattern variable may be present in the pattern (so in the second argument of `mostInfo`). This pattern variable is the one specified in the first argument. Thus, in this case, the information in the first argument is superfluous. Nevertheless, it is has been added to take into account future extensions of Plosequamo-language in which it may be sensible to indicate which pattern variables you want to target.

The syntax of Plosequamo-language is defined as follows.

PlosequamoFormula



⁶This definition does not cover certain cases: there are examples in which two deducible sentences exist that both match the pattern, but that are independent (so, not deducible) from each other, and which have a different size for `s_`. This means that none of these sentences is 'more informative' than the other. With thanks to Jesse Maas (also known as 'Aeduin') for discovering this. This problem can be solved easily, but this is left as future work.

III.8.5.6 Folminquon-language

Folminquon-language

Folminquon-language (first-order logic minus quantifiers and connectives) is a fragment of first-order logic with equality and inequality. It does not contain quantification or connectives. (No connectives means the absence of: negation, implication, disjunction and conjunction.) The absence of quantification implies the well-formed formulas of the language do not contain variables, only constants. The following examples together cover the complete expressivity of *Folminquon-language*:

P(akwasi, john)
Binary predicates.

MusicTrio(corea, cohen, ballard)
3-place predicate. <i>Folminquon-language</i> allows any arity.

AllUnequal(akwasi, john, ebere)
Inequality: akwasi, john and ebere refer to mutually distinct entities. AllUnequal is not a predicate, but syntactic sugar to express a number of inequality sentences at once, in this case: (not Equal(akwasi, john)) and (not Equal(akwasi, ebere)) and (not Equal(john, ebere)).

AllEqual(akwasi, adrian, brandon)
Equality: akwasi, adrian and brandon refers to the same entity. AllEqual is not a predicate, but syntactic sugar to express a number of equality sentences at once, in this case: equal(akwasi, adrian) and equal(adrian, brandon).

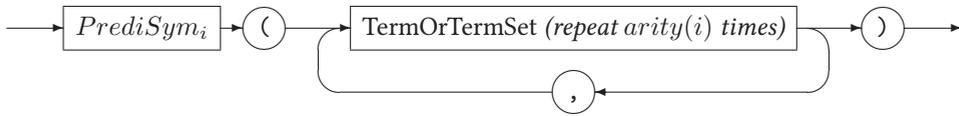
P(akwasi, {john, fiona})
Syntactic sugar to express P(akwasi, john) and P(akwasi, fiona) at once. Thus, in general, this construct is an abbreviation for a collection of predicate sentences: all predicate sentences that can be formed by selecting exactly one constant from the sets that occur in the construct. I introduce the name term-set for such a set. Term-sets, as in the example above, are enclosed in “{}”.

The syntax of *Folminquon-language* is defined as follows.

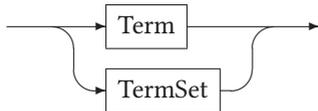
FolminquonFormula



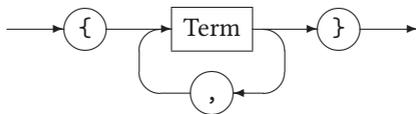
PredicateApplication



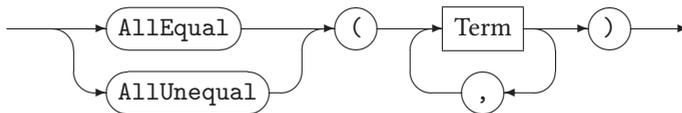
TermOrTermSet



TermSet



EqualityFormula



where all non-terminals not defined in the diagram above, are as defined in previously occurring syntax diagrams.

The semantics of Folminquon-language is formally defined by translation to FOL:

$$T_{\text{folminquon}}(\varphi) = \varphi \text{ iff } \varphi \text{ does not contain EqualityFormulas.} \quad (\text{III.8.6})$$

$$T_{\text{folminquon}}(\text{AllEqual}(c_1, \dots, c_n)) = \text{Equal}(c_1, c_2) \text{ and Equal}(c_2, c_3) \dots \text{and Equal}(c_{n-1}, c_n) \quad (\text{III.8.7})$$

$$T_{\text{folminquon}}(\text{AllUnequal}(c_1, \dots, c_n)) = (\text{not Equal}(c_1, c_2)) \text{ and } (\text{not Equal}(c_1, c_3)) \dots \text{and } (\text{not Equal}(c_{n-1}, c_n)), \quad (\text{III.8.8})$$

For all clarity, the right-hand side of (III.8.8) contains a (not Equal . . .)-formula for *all* possible pairs of constants occurring at the left-hand side.

The translation of sentences that contain term-sets are left to the reader.

III.8.5.7 Folnuminquon-language

Folnuminquon-language (*Folnuminquon-language* (FOL with number-quantifiers, minus quantifiers and connectors) is an extension of *Folminquon-language*: it allows one to make statements about the number of individuals satisfying a certain formula. These are therefore extensions of the ordinary existential quantifier. (Note that they are, therefore, an instance of generalised-quantifiers (Westerståhl, 2016).) Therefore, note that *Folnuminquon-language* is, syntactically speaking, not a fragment of FOL, because FOL does not allow such constructs. However, semantically speaking *Folminquon-language* is a fragment of FOL, because all *Folminquon-language* can be translated to FOL sentences with an equivalent meaning.

Examples of *Folnuminquon-language* sentences are the following:

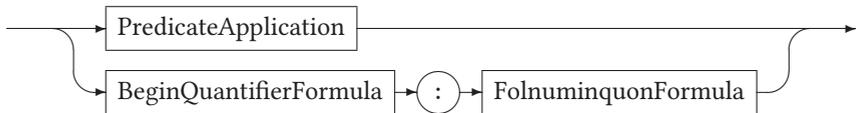
$=4x:P(c2, x)$
There are exactly 4 distinct individuals x for which holds $P(c2, x)$.

Other examples with an analogous meaning can be obtained by using: >5 , <3 , $=<11$ (“less than or equal”) and $>=1$ (“greater than or equal”) instead of $=4$ in the example above.

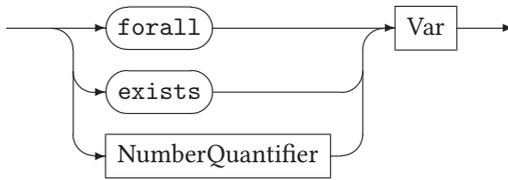
The syntax of *Folnuminquon-language* is defined by the following replacements in the syntax diagram of *Folminquon-language*:

- (1) Replacement of *FolminquonDocument* with *FolnuminquonDocument*.
- (2) Replacement of all occurrences of the word *Folminquon* with *Folnuminquon*.
- (3) Adding the following rules to the syntax diagram of *Folminquon-language* (see III.8.5.6):

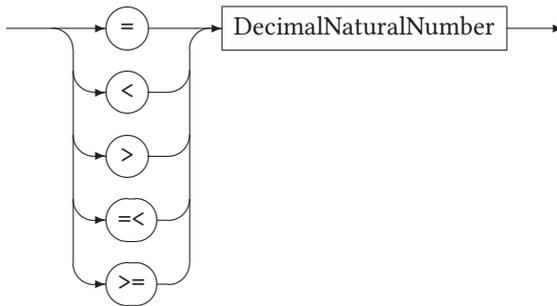
FolnuminquonFormula



BeginQuantifierFormula



NumberQuantifier



where *DecimalNaturalNumber* is a decimal representation of a natural number, including 0.

The semantics of the language is defined by means of translation to FOL. The translation function $T_{folnuminquon}$ from FOL sentences to Folnuminquon-language sentences is inductively defined as follows:

$$T_{folnuminquon}(\varphi) = \varphi \text{ iff } \varphi \text{ does not contain number-quantifiers.} \quad (\text{III.8.9})$$

$$T_{folnuminquon}(\geq_n \mathbf{x} : \varphi(\mathbf{x})) = \text{inequalityStats and } T_{folnuminquon}(\varphi(\mathbf{x}_1)) \text{ and } T_{folnuminquon}(\varphi(\mathbf{x}_2)) \text{ and } \dots \text{ and } T_{folnuminquon}(\varphi(\mathbf{x}_n)), \quad (\text{III.8.10})$$

where

$$\begin{aligned} \text{inequalityStats} = & \text{exists } x_1 : \text{exists } x_2 : \dots \text{exists } x_n : \\ & x_1 \neq x_2 \text{ and } x_1 \neq x_3 \text{ and } \dots \text{ } x_1 \neq x_n \\ & x_2 \neq x_3 \dots x_{(n-1)} \neq x_n \quad (\text{III.8.11}) \end{aligned}$$

The other number-quantifiers can now be defined in terms of the number-quantifier defined above.

$$T_{\text{folnuminquon}}(<n \ x:\varphi(x)) = \text{not } T_{\text{folnuminquon}}(>=n \ x:\varphi(x)) \quad (\text{III.8.12})$$

$$T_{\text{folnuminquon}}(= <n \ x:\varphi(x)) = T_{\text{folnuminquon}}(<(n+1) \ x:\varphi(x)), \quad (\text{III.8.13})$$

$$T_{\text{folnuminquon}}(=n \ x:\varphi(x)) = T_{\text{folnuminquon}}(>=n \ x:\varphi(x)) \text{ and } T_{\text{folnuminquon}}(<=n \ x:\varphi(x)) \quad (\text{III.8.14})$$

III.8.5.8 Pattern-languages Plonu-language and Plonumo-language

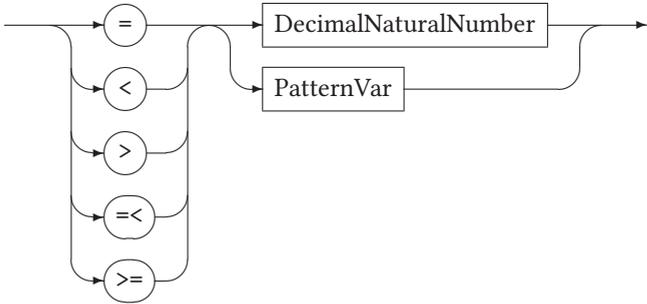
Plonu-language *Plonu-language* is a pattern-language for Folnuminquon-language. In it, the numbers in the number-quantifiers may be replaced with pattern variables.

$>n_ \ x:P(c2, x)$
Pattern representing the set of sentences in Folnuminquon-language that can be obtained by substituting the pattern variable $n_$ by a number.

The syntax of Plonu-language is defined by the following replacements in the syntax diagram of Folnuminquon-language (see section III.8.5.4 (p. 270)):

- (1) Replacement of all occurrences of the word *Folnuminquon* with *PlonuFormula*.
- (2) Replacement of *NumberQuantifier* of the syntax diagram of Folnuminquon-language with the following production rules:

NumberQuantifier



This, however, is not sufficient. Suppose, for example, that someone poses the question how many children Akwasi at least has. In the context of English, and probably most other natural languages, that person is not asking for any value that is correct, but for the value with the highest information content that can be determined based on the translation. *Plonumo-language*, an extension of *Plonu-language*, solves

Plonumo-
language

this problem. It contains an additional `mostInfo`-construct. Examples follow.

<code>mostInfo(n_, >=n_...)</code>
Select the most informative sentence(s) from all deducible sentences that match the pattern: in this case the sentence with the <i>greatest</i> value for <code>n_</code> .

<code>mostInfo(n_, =<n_...)</code>
Select the most informative sentence(s) from all deducible sentences that match the pattern: in this case the sentence with the <i>smallest</i> value for <code>n_</code> .

Omit the '=' for the rest of the definitions.

The syntax of Plonumo-language is defined as follows.

PlonuFormula



III.8.5.9 Implementation of a reasoner for Plonumo-language

Realising a reasoner for the relatively simple language Plonumo-language turned out not to be trivial. At first glance, I assumed that there would be an off-the-shelf reasoners that would be able to do this out-of-the-box. (Obviously, with out-of-the-box I mean that it at most requires the creation of a trivial algorithm to translate the Plonumo-language sentence to the query-language of the given reasoner.) Unfortunately, to the best of my current knowledge, such a reasoner does not exist.

Moreover, during my investigations to implement a dedicated reasoner from scratch, I was able to define a reduction of the graph colouring problem to an important part of the problem posed in Plonumo-language. The graph colouring problem is NP complete, so this also holds for Plonumo-language. For me, it was somewhat surprising that a problem that appeared to me to be quite simple can only be solved with a brute-force algorithm (provided that $P \neq NP$).

Finally, I implemented a reasoner for a part of Plonumo-language by stitching together three components: Eprover, a small algorithm implemented by me and Paradox⁷. The reasoner is limited to Plonumo-language sentences in which `>=` occur, for example

```
mostInfo( n_, >=n_ x:hasTeacher(selinde, x) ).
```

⁷My gratitude goes to Geoff Sutcliffe who made me aware of Paradox.

This is the only part of Plonumo-language needed in one of the SWiFT-Focused instances, to wit SWiFT-NoUna. The basic strategy is as follows:

1. Let K be the KR-base in Folnuminquon-language, and φ be the reasoning-goal in Plonumo-language, which has \geq as its comparison operator. *Example:* $\varphi = \text{mostInfo}(n_-, \geq n_- x:\text{hasChild}(c, x))$.
2. Create a copy of K : K' . Eliminate the equality and inequality sentences from K' . Translate K' to TPTP language. This results in K'_{TPTP} .
3. Let φ' be a copy of φ without the `mostInfo`-construct, and without the number-quantifier, and with the variable transformed into a pattern variable. *Example:* $\varphi' = \text{hasChild}(c, x_-)$. Translate φ' into TPTP language: φ'_{TPTP} .
4. K'_{TPTP} and φ'_{TPTP} are consequently offered to Eprover. Eprover produces the set of constants for which φ'_{TPTP} holds: C_{eprover} .

Subsequently, the second stage takes care of the `mostInfo`-construct and the number-quantifier in φ . The goal is to produce the largest number for which can be deduced that φ' holds. Because of the absence of the unique name assumption, this number is not equal to the number of constants in C_{eprover} . After all, some of these constants may be referring to the same entity. For this purpose, the Plonumo-language-reasoner has to deal with the equality and inequality sentences. First, C_{eprover} has to be transformed. The reason is that some (in)equality sentences in K may contain constants that are not in C_{eprover} . However, simply omitting these sentences is incorrect. Indirectly, such sentences may express information about the equality or inequality of constants within C_{eprover} . This is the ‘non-trivial’ additional reasoning that has to take place next to the reasoning provided by Eprover and Paradox. The following code specifies it in pseudo code:

```
EqualityEliminator(ConstantSet, EqualityAndInequalitySentences) =
{ InequalitySentences := EqualityAndInequalitySentences
  for each Equal(a,b) in InequalitySentences do
    { if a is in ConstantSet then
      { relabel all constants b in InequalitySentences to a }
      else if b is in ConstantSet then
      { relabel all constants a in InequalitySentences to b }
      else
      { relabel all constants b in InequalitySentences to a }
      (endif)

      eliminate Equal(a,b) from InequalitySentences
    }
}
```

```
return InequalitySentences
}
```

In the final stage, Paradox is applied:

1. Create a copy of K that only contains the equality and inequality sentences: K_{EqIneq} .
2. Apply EqualityEliminator to K_{EqIneq} :
 $K_{Ineq} = \text{ConstantsTransformation}(C_{\text{eprover}}, K_{EqIneq})$. Note that K_{Ineq} does not contain any Eq sentences.
3. Drop all inequality statements from K_{Ineq} that contain at least one constant that is not in C_{eprover} : K_{IneqC} .
4. Apply Paradox to K_{IneqC} .

I implemented this strategy in Scala. The code below presents a snippet that forms the core of the implementation. The full source code can be accessed in the SWiFT-Focused repository

(<https://github.com/swiftgame/SWiFTfososc>)

at

<https://github.com/swiftgame/SWiFTfososc/blob/develop/webApp/swift/src/main/scala/org/ocbk/swift/reas/tpfolnuminqua.scala>

Note that the identifiers used in the source code are not necessarily the same as the ones used in the reasoning strategy described above.

```
// Omitted from this fragment: import-statements
```

```
object Prover
{ def query(query:FolnuminquaQuery, ft:FOLtheory):Int =
  { /* Only what matches is currently supported. Thus, warnings that the
    match statement is not exhaustive is a consequence of a
    deliberate choice.
    */
    println("#### incoming FOLtheory:\n" + ft)
    val ft_noEqStats = ft.copy
    ft_noEqStats.removeStats
    ( { case stat:Equal => true
      case stat:Unequal => true
      case _ => false
```

```

    }
  )
  println("\n#### (in)eq stats removed:\n" + ft_noEqStats)
  //println(" incoming theory must stay unaffected:\n" + ft)
  var ft_noEqStats_fof = ft_noEqStats.exportToTPTPfof

  // translate query to fof
  val queryFof:String =
  query match
  { case Sharpest( NumResPat(Geq, PatVar(numpatvarname),
                          Var(boundvarname), PredApp(p,consts)))
    =>
    { val pv = "PV" + boundvarname
      "fof(form" + ft_noEqStats.stats.length + ", question,
      ? [" + pv + "] : " + p.name +
      consts.map
      ( c =>
        if( c.name.equals(boundvarname) )
          pv
        else
          c.name
        ).mkString(",","") + ")."
      }
    }
  }
  ft_noEqStats_fof += queryFof
  println("\n#### translated to fof and added query in fof format:\\
\n" +
         ft_noEqStats_fof)

  // write to file
  var outFile = new File("ft_noEqStats.fof")
  var fullpath = outFile.getAbsolutePath
  println("\n#### creating file: " + fullpath)
  var out:PrintWriter = new PrintWriter(new BufferedWriter(
    new FileWriter(outFile)))
  out.print(ft_noEqStats_fof)
  out.flush
  out.close

  // apply eprover
  val eproverResult = Eprover(
    "--cpu-limit=30 --memory-limit=Auto --tstp-format -s \\
--answers "

```

```

        + fullpath
    )

println("#### eprover's result =\n" + eproverResult)
val c:List[Constant] = eproverResult.extractConstants
println("    extracted constants = " + c)
if(c.length != 0) /* needed, because Paradox will produce model size
                  = 1 for a theory which only consists of
                  fof(form0, , introconstants())
                  */
{ /* eliminate equality statements with constants in c as preferred
   constants
   */
  println("\n#### start eliminate equality statements")
  val ft_EqInEqIntroConsStats = ft.copy

  // { Paradox needs a statement that introduces the constants
  val pred = Predicate("introduceconstants", c.length)
  // }

  ft_EqInEqIntroConsStats.removeStats
  ( { case stat:Equal => false
      case stat:Unequal => false
      case _ => true
    }
  )
  println("    create a statement to introduce all constants from \\  

C and transform to TPTP fof")
  ft_EqInEqIntroConsStats.addStat(PredApp_FOL(pred, c))
  println("    first drop everything but (in)equality stats:")a
  println("    ft_EqInEqIntroConsStats = "  

    + ft_EqInEqIntroConsStats)
  println("    applying EqualityEliminator:")
  val ft_onlyInEqIntroConsStats =
    EqualityEliminator(ft_EqInEqIntroConsStats, c)

  println("\n#### drop all inequality statements which contains \\  

at least one constant which is NOT in C.")
  def f(cs:List[Constant], stat:FOLstatement):Boolean =
  { stat match
    { case Unequal(c1, c2) => !cs.contains(c1) || !cs.contains(c2)
      case _ => false
    }
  }

```

```

    }

    ft_onlyInEqIntroConsStats.removeStats(f(c,_))
    println("  ft_onlyInEqIntroConsStats becomes "
           + ft_onlyInEqIntroConsStats
           )
    println("\n#### apply paradox")
    val ft_onlyInEqIntroConsStats_fof =
      ft_onlyInEqIntroConsStats.exportToTPTPfof
    println("  ft_onlyInEqIntroConsStats_fof = "
           + ft_onlyInEqIntroConsStats_fof)
    // { write to file
    outFile = new File("ft_onlyInEqIntroConsStats.fof")
    fullPath = outFile.getAbsolutePath
    println("\n#### creating file: " + fullPath)
    out = new PrintWriter(new BufferedWriter(new FileWriter(outFile)))
    out.print(ft_onlyInEqIntroConsStats_fof)
    out.flush
    out.close
    // }

    val paradoxResult = Paradox("--model --verbose 0 " + fullPath)
    println("  result of paradox:\n" + paradoxResult)

    paradoxResult.getModelSize
  }
  else 0
}
}
}

```

The Scala implementation of EqualityEliminator is as follows. It can be found in the SWiFT-Focused repository at the following URL:

```

https://github.com/swiftgame/SWiFTfososc/blob/develop/webApp/swift/src/main/scala/org/ocbk/swift/reas/equalityEliminator.scala

```

```

object EqualityEliminator
{
  def apply(ft:FOLtheory, prefCs:List[Constant]):FOLtheory =
  {
    /* first remove equality statement from ft, and then do the constant
       replacement in the rest.
    */
    println("eliminateEqualities")
    println("  ft was:" + ft)
  }
}

```

```

ft.stats.find( { case s:Equal => true; case _ => false } ) match
{
  case None => { println("  ft becomes:" + ft); ft }
  case Some(Equal(a,b)) => { ft.removeStat(Equal(a,b))
    if (prefCs.contains(a))
      ft.substituteConstant(b,a)
    else if (prefCs.contains(b))
      ft.substituteConstant(a,b)
    else ft.substituteConstant(b,a)

    println("  ft becomes:" + ft)
    apply(ft, prefCs)
  }
}
}
}
}

```

III.8.5.10 Bridge-language Basic-language

Basic-language *Basic-language* (BRidge language bASIC) is a basic bridge-language that can be used for any of the formal-languages introduced in this work that has constants and unary predicates with semantic conditions as used in FOL. Thus, each constant refers to a unique entity, and each unary predicate to a unique unary relation in the domain of discourse. It forms a bridge between the constants and predicates on the one hand, and the corresponding words used in natural language on the other hand.

EntityBridge(e, {"Edward"})
The formal-language constant e refers to the same entity as the natural language word "Edward".

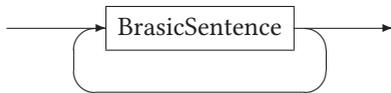
EntityBridge(e, {"Edward", "Akwasi"})
The formal-language constant e refers to the same entity as the natural language words 'Edward' and 'Akwasi'. (In other words, 'Edward' and 'Akwasi' are two names for the same entity).

UnaryPredicateBridge(B, {"big"})
The meaning of formal language predicate B corresponds with the meaning of the natural language word 'big'. An example: with the information expressed above: B(e) translates into natural language as follows: "[The translation into natural language of e] is big". (The translation is partial, because in this case, no Basic-language sentence for e is given.)

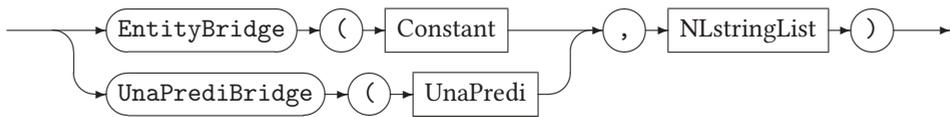
EntityBridge(e, {"Edward", "Akwasi"})
UnaryPredicateBridge(B, {"big"})
Given these bridge sentences, the FOL sentence B(e) can be translated into natural language as follows: "Edward is big" or "Akwasi is big".

The syntax definition of Basic-language is as follows.

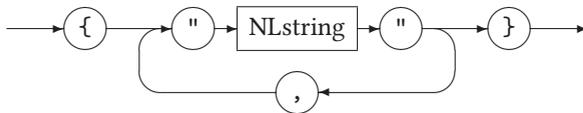
BasicDocument



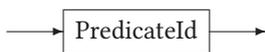
BasicSentence



NLstringList



UnaryPredicate



PredicateId



where *UpperCaseId* is as defined as in previous diagrams, and *UnaPrediBridge* stands for *UnaryPredicateBridge*, and *UnaPredi* for *UnaryPredicate*.

III.8.5.11 Hurelan-language

Hurelan-language

Hurelan-language (HUMAN RELational antonyms LANguage) is a bridge-language that can be used to extend an existing formal-language that allows binary predicates. This work introduces it for the purpose of enabling (natural language) text generation of certain SWiFT-Focused challenges.

relational-antonym

Relational antonyms, a term from linguistics, are pairs of words that refer to a relationship from opposite points of view, such as parent/child or borrow/lend (Plag, 2007).

human-relational-antonym

This work additionally introduces the notion *human-relational-antonym*: a relational antonym in which the words express roles that can be fulfilled by humans, such as parent/child, teacher/student, husband/wife.

hurelan-construct

The *hurelan-construct* enables one to express (1) that a binary predicate of a formal-language expresses a human-relational-antonym, and (2) the associated words in natural language.

Examples follow.

```
Hurelan(hasChild, "parent", "child")
```

The binary predicate `hasChild` expresses a relation between its arguments, that can be expressed with human relational antonyms in natural language. If translated into natural language, the role of the first argument of `hasChild` is expressed as "parent" and the role of the second as "child".

```
Hurelan(hasChild, "parent", "child")
```

```
EntityBridge(a, {"Akwasi"})
```

```
EntityBridge(j, {"Jules"})
```

```
hasChild(a, j)
```

The Hurelan-language statement and the bridge statement provide the information that is needed to be able to translate the last sentence (automatically) to natural language: "Akwasi is a parent of Jules.", or "Jules is a child of Akwasi."

III.8.6 SWiFT·Efe

SWiFT·Efe focuses on the basic understanding of constants and predicates. It is intended to be playable by novices. 'Efe' is a Nigerian (Igbo) word that means 'easy' or 'simple'.

III.8.6.1 The challenge-set

The challenge-set of SWiFT·Efe consists of natural language sentences exemplified by the following:

- “Akwasi is big.”
- “Soerdjaja is fast.”
- “Soerdjaja, Zwirdi and Pompi are fast.”
- “Zwobo is fast and big.”
- “Darpo and Yuna are fast and big.”

For each text from Efe’s challenge-set, it is reasonable to assume that probably all English speakers will assume that each entity is referred to with at most one proper name. For example, in the last example, ‘Darpo’ and ‘Yuna’ are names of two distinct persons, and not two names of one person. (The reader is invited to see whether (s)he finds it intuitive that a sentence from this set, provided in isolation, would suggest otherwise.) So when constructing a bridge, it is not needed to assume that a player has to connect more than one proper name to a constant.

III.8.6.2 Formal-languages

III.8.6.2.1 Formal-target-language: Efe-language

The formal-target-language is Efe-language, as defined in section III.8.5.3 (p. 270).

III.8.6.2.2 Text-generation-language

The text-generation-language is equal to the formal-target-language (Efe-language).

III.8.6.2.3 Answer-language: Folsequa-language

The answer-language of SWiFT·Efe is Folsequa-language, as defined in section III.8.5.4 (p. 270).

III.8.6.2.4 Question-formal-language: Plosequamo-language

SWiFT·Efe’s question-formal-language is Plosequamo-language. Given the questions that are generated by the computer, only a fragment of Plosequamo-language is actually used.

III.8.6.2.5 Bridge-language: Brasic-language

The bridge-language of SWIFT-Efe is Brasic-language, as defined in section III.8.5.10 (p. 286).

III.8.6.3 The question set

The following provides examples of questions that can be generated by the computer, together with their representation in the question-formal-language (Plosequamo-language):

Q1 "Mention all people who are big based on the information in the text."
`mostInfo(s_, forall x from s_:B(x))`

Q2 "Mention all people who are big and fast based on the information in the text."
`mostInfo(s_, forall x from s_:(F(x) and B(x)))`

Note that with these questions the player does not need insight in the absence of the unique name assumption, because there are no equality axioms. The only way to be certain that all people are mentioned is by mentioning all constants with the F predicate (even if some may refer to the same individual).

III.8.6.4 Typical session example

This section contains a typical example of a SWIFT-Efe playing session. The events are shown in the order at which they occur, however, in some cases they are shown to the player in another order.

III.8.6.4.1 Computer-generated challenge in Efe-language and Brasic-language (not shown to player)

Representation in Efe-language:

```
F(akwasi)
F(john)
F(szymon)
```

Bridge sentences in Brasic-language:

```
EntityBridge(akwasi, {"Akwasi"})
EntityBridge(john, {"John"})
EntityBridge(szymon, {"Szymon"})
```

III.8.6.4.2 Computer-generated challenge in natural language (shown to player)

“Akwasi, John and Szymon are big.”

III.8.6.4.3 Player’s translation in Efe-language and Basic-language

Representation in Efe-language:

F(a)

F(j)

F(s)

Bridge sentences in Basic-language:

EntityBridge(a, {"Akwasi"})

EntityBridge(j, {"John"})

EntityBridge(s, {"Szymon"})

Note: during experimentation, I typically observed that players start abbreviating longer names after having played a few sessions, just as in the above example.

III.8.6.4.4 Computer-generated question-attack-round in Plosequamo-language (not shown to player)

mostInfo(s_, forall x from s_:B(x))

III.8.6.4.5 Computer-generated question-attack-round in natural language (shown to player)

“Mention all people who are big based on the information in the text.”

III.8.6.4.6 Computer-generated algorithmic-defence-round (shown to player)

The algorithmic-defence-round consists of the stages covered in the following two subsections.

III.8.6.4.7 Computer-generated algorithmic construction stage (shown to player)

The algorithmic construction stage is based on the computer-generated question-attack-round. As a consequence of the choices for the formal languages in SWiFT-Efe the two are identical:

```
mostInfo(s_, forall x from s_:B(x))
```

III.8.6.4.8 Computer-generated algorithmic execution stage (shown to player)

Plosequa-language: forall x from {akwasi, john, szymon}:B(x)
 natural language: "Akwasi, John and Szymon are big."

Deduced from the player-created translation:

Plosequa-language: forall x from {a, j, s}:B(x)
 natural language: "Akwasi, John and Szymon are big."

Using the bridge-sentences of both translations, the implemented algorithm will now automatically deduce that both answers are identical. Note that this also works if the mentioned entities have a different order in the computer-generated formal language representation than in the player-created translation.

III.8.6.4.9 Conclusion (shown to player)

The computer communicates to the player that the translation is correct.

III.8.7 SWiFT-NoUna

NoUna

SWiFT-NoUna focuses on the understanding of the absence of the unique name assumption. *NoUna* stands for 'No Unique Name Assumption'. The unique name assumption is the assumption that each constant refers to a distinct entity, or, equivalently, that there are no constants that refer to the same entity (Russell and Norvig, 2016). This is an assumption often tacitly held by people, while this is not necessarily the case for formal languages. This is a well-known source of faulty representations in, among others, the field of knowledge engineering.

The challenge-set, question set and formal-target-language are chosen such that the probability is very high that the player will create a faulty translation when he or she is not aware of the unique name assumption. This is achieved by presenting a natural language text in which a number of named individuals share a certain property. Without the unique name assumption, the translation of these names to constants, *should* be accompanied by inequality sentences. If these sentences are omitted, the answer to questions about the number of individuals with the given property will be wrong, because the reasoner cannot determine whether some of the constants refer to the same, or distinct individuals.

III.8.7.1 The challenge-set

The challenge-set of SWiFT·Efe consists of natural language sentences exemplified by the following:

- “Ebere is a trainer of Eva.”
- “Selinde is a grandchild of Ebere and Frank.”
- “Alexia is a grandparent of Pondiwu, Gaston and Kevin.”

III.8.7.2 Formal-languages

III.8.7.2.1 Formal-target-language: Folminquon-language

The formal-target-language is Folminquon-language, as defined in section III.8.5.6 (p. 274).

III.8.7.2.2 Text-generation-language: Folminquon-language + Hurelan-language

The text-generation-language of SWiFT·NoUna is equal to Folminquon-language (section III.8.5.6 (p. 274)) extended with the hurelan-construct (section III.8.5.11 (p. 288)).

III.8.7.2.3 Answer-language: Folnuminquon-language

The answer-language of SWiFT·NoUna is Folnuminquon-language, as defined in section III.8.5.7 (p. 276).

III.8.7.2.4 Question-formal-language: Plonumo-language

The question-formal-language of SWiFT·NoUna is Plonumo-language (section III.8.5.8 (p. 278)).

III.8.7.2.5 Bridge-language: Brasic-language + Hurelan-language

The bridge-language of SWiFT-NoUna is Brasic-language, as defined in section III.8.5.10 (p. 286), extended with the hurelan-construct. The latter allows the translation of predicates that express a human relational antonym.

III.8.7.3 The question set

The following provides the questions that can be generated by the computer, together with their representation in the question-formal-language (Plonumo-language), and the required bridge-language statements:

Q1 “How many children does Akwai minimally have? Give the highest number for which this can be established with certainty.”

```
EntityBridge(akwasi, {"Akwasi"})
Hurelan(hasChild, "parent", "child")

mostInfo( n_, >=n_ x:hasChild(akwasi, x) )
```

III.8.7.4 Typical session example

III.8.7.4.1 Computer-generated challenge in Folminquon-language + Hurelan-language and Brasic-language (not shown to player)

Representation in Folminquon-language:

```
hasTeacher(selinde, {ebere, frank})
AllUnequal(selinde, ebere, frank)
```

Bridge sentences in Brasic-language and Hurelan-language:

```
Hurelan(hasTeacher, "student", "teacher")
EntityBridge(selinde, {"Selinde"})
EntityBridge(ebere, {"Ebere"})
EntityBridge(frank, {"Frank"})
```

III.8.7.4.2 Computer-generated challenge in natural language (shown to player)

“Selinde is a student of Ebere and Frank.”

III.8.7.4.3 Player's translation in Efe-language and Brasic-language

Representation in Folminquon-language:

```
hasTeacher(sel, {ebe, fra})
```

Bridge sentences in Brasic-language and Hurelan-language:

```
Hurelan(hasTeacher, "student", "teacher")
EntityBridge(sel, {"Selinde"})
EntityBridge(ebe, {"Ebere"})
EntityBridge(fra, {"Frank"})
```

As can be seen in this example, a typical error is made by the player: unaware of the absence of the unique name assumption, the player did not add the necessary inequality axioms.

III.8.7.4.4 Computer-generated question-attack-round in Plonumo-language (shown to player)

```
Plonumo-language: mostInfo( n_, >=n_ x:hasTeacher(selinde, x) )
```

(Select the most informative sentence(s) from all deducible sentences that match the pattern: in this case the sentence $\geq n_ x:hasTeacher(selinde, x)$ with the highest value for $n_.$)

III.8.7.4.5 Computer-generated question-attack-round in natural language (shown to player)

“How many teachers does Selinde minimally have? Give the highest number for which this can be established with certainty.”

III.8.7.4.6 Computer-generated algorithmic-defence-round (shown to player)

The algorithmic-defence-round consists of the stages covered in the following two subsections.

III.8.7.4.7 Computer-generated algorithmic construction stage (shown to player)

The algorithmic construction stage is based on the computer-generated question-attack round. As a consequence of the choices for the formal languages in SWiFT-NoUna the two are identical:

Plonumo-language: `mostInfo(n_, >=n_ x:hasTeacher(selinde, x))`

III.8.7.4.8 Computer-generated algorithmic execution stage (shown to player)

Deduced from the computer-generated ‘translation’ of the challenge text:

Folnuminquon-language: `>=2_ x:hasTeacher(selinde, x)`

natural language: “Selinde has minimally 2 teachers, and this is the highest number for which this can be established with certainty.”

Deduced from the player-created translation:

Folnuminquon-language: `>=1_ x:hasTeacher(selinde, x)`

natural language: “Selinde has minimally 1 teacher, and this is the highest number for which this can be established with certainty.”

III.8.7.4.8.1 Analysis The answer of the player is wrong as a consequence of the omission of inequality axioms.

III.8.7.4.9 Conclusion (shown to player)

The computer communicates to the player that the translation is incorrect (without stating why).

III.8.8 Conclusion and future work

III.8.8.1 Conclusion

The research questions highlighted in the introduction of this chapter (section III.8.1.1 (p. 246)) and a summary of the corresponding research results achieved in this chapter are as follows.

“

Research question 2.2.1 (incentives SWiFT-game). *Which incentives for participating and improving formal-fluency does SWiFT provide, and how could the game be improved to make these incentives stronger?*

(Quoted from page 221.)

”

Conclusion 2 of research question 2.2.1. *The incentives SWiFT-Focused provides in comparison with SWiFT-Fixtal and SWiFT-Full include that it can be played in a much shorter time, and that it is far easier to learn how to play the game because the player only has to fulfill one role. This is further addressed in the conclusion 1 of research question 2.2.4 on the following page and the conclusion 1 of research question 2.2.6 (p. 299).* ■

“

Research question 2.2.2 (easier comparison translations). *easier comparison translations*

(Quoted from page 221.)

”

Conclusion 2 of research question 2.2.2. *In SWiFT-Focused the comparison of the translations' quality is greatly simplified. The essential cause is that each SWiFT-Focused-instance has, in several respects, a much narrower scope than SWiFT-Full and SWiFT-Fixtal. Just as with SWiFT-Fixtal, the formal-target-language is fixed, which yields the same advantages as mentioned in SWiFT-Fixtal (see conclusion 1 of research question 2.2.2 (p. 242)).*

Additionally, the text comes from a much narrower set of possible texts. The questions in the question-attack-round come from a narrower set, too. They do not include all possible questions about the texts, but are restricted to questions that test the subcapability that is in focus. The narrowness is chosen such that a computer can automatically generate the texts and the questions for the question-attack-round. The text, moreover, is quite short. This has all been explained in section III.8.1.2 (p. 246).

The reasons that this makes it easier to reliably compare the translations include: (1) The automated generation gives a great level of control over the quality of the questions, and, hence, the comparison of the quality of the translations. (2) The narrower set of possible questions and shorter texts means that it is easier to check to what extent it is possible to automatically deduce from the translation all sentences that can

be deduced from the original text. To be precise: it is easier to approximate the metric ‘automatically-deducible-fraction’ (see section III.6.4.1 (p. 218)).

Note that SWiFT-Focused sacrifices testing full-blown formal-fluency for this, therefore, it is not a substitute for SWiFT-Focused but a complementation it (see). ■

“

Research question 2.2.3 (no immediate full-blown formal-fluency needed).

How to design a SWiFT-game in which the player is not immediately confronted with acquiring full-blown formal-fluency, but which still has many to most of the desired effects of SWiFT-Full?

(Quoted from page 222.)

”

Conclusion 2 of research question 2.2.3. *In the game variant developed in this chapter, higher-order-SWiFT, the full challenge of mastering formal-fluency is broken up into smaller subchallenges. Each subchallenge focuses on a specific problem within formal-fluency, and is presented to the player in a mini-game, coined a SWiFT-Focused game. In this single-player game, a player is offered a text to translate, after which the translation is tested by comparing what can be deduced from it with reasoners with what can be deduced from the original natural language text. SWiFT-Focused games that cover interdependent subchallenges are presented to the player in order of increasing level of difficulty. Each SWiFT-Focused-game is a single player game, in which the player only fulfills the role of translator from natural language. All other roles are fulfilled by the computer. This section already presented the design of three SWiFT-Focused games, which are also implemented. I conclude that both on logical grounds and preliminary experimentation, higher-order-SWiFT is a promising answer to the mentioned disadvantages.* ■

“

Research question 2.2.4 (reduced human-resources). *How to design a SWiFT-game or variant thereof that is effective for fostering formal-fluency, but that is less human-resource intensive than SWiFT-Full?*

(Quoted from page 222.)

”

Conclusion 1 of research question 2.2.4. *The solution presented in this chapter is a direct consequence of higher-order-SWiFT, an extension of higher-order-Orcoba.*

Higher-order-SWiFT allows the challenge of formal-fluency to be broken up in small subcapabilities. The break-up can be chosen such that the translation from natural language to formal-language can be done automatically. In this way, the players have to translate only the text. All other roles can be fulfilled by the computer, which dramatically reduces human resource usage. More detailed information has been presented in section III.8.1.2 (p. 246). ■

“

Research question 2.2.6 (short games). *How to design a SWiFT-game in which playing a game does not take much time for the player? (That means, playing a game session takes a few seconds to at most a few minutes, and learning to play the game not more than five to ten minutes.)*

”

(Quoted from page 242.)

Conclusion 1 of research question 2.2.6. *There are several parts to the result. These are all directly or indirectly related to the fact that in Higher-order-SWiFT the overall challenge of fostering formal-fluency is broken up into smaller subchallenges that are addressed by suitably designed SWiFT-Focused games. The parts to the result are as follows.*

- *The subcapabilities in higher-order-SWiFT are – as a requirement – chosen such that they can be adequately covered by a composition-record that has a pre-defined maximum size. Choosing a sufficiently low maximum size allows sufficiently reducing the average maximum playing time needed for this round.*
- *Moreover, the break-up into subcapabilities is required to be done such that challenge-sets can be created that allow a sufficient reduction in the average maximum time needed to play the translation round.*
- *In higher-order-SWiFT the formal-fluency player only has to fulfill one role, that of a translator (see conclusion 1 of research question 2.2.4 on the facing page). This makes it far easier to learn how to play the game.*

■

“

Research question 2.2.7 (lessons general reasoner construction). *What are lessons beyond the scope of SWiFT-Focused that can be learnt regarding the effective design of automated deductive reasoners?*

”

(Quoted from page 246.)

Conclusion 1 of research question 2.2.7. *The design of a reasoner requires the specification of a KR-language and a query-language. I presented an approach that unifies querying with knowledge representation: by defining a sufficiently expressive language that allows one to express the structure the answer should meet. I believe this is, in several respects, far more elegant and practical than somehow formalising natural language questions, or working with a collection of ad-hoc query language constructs. In the approach, the design of a query-language is as follows. First, a formal-language, the answer-language, is defined that is an extension of the KR-language of the reasoner. This is crucial to reach the desired expressivity. Second, a language is defined that allows specifying the structure the answer (from the answer-language) should meet. This language is used as the query-language. A structural and elegant way to do this, is by taking the answer-language as the basis of a pattern-language.* ■

“

Research question 2.2.4 (performance metric). *What is a good performance metric for the performance of the players of the SWiFT-Focused games?*

”

(Quoted from page 259.)

Conclusion 1 of research question 2.2.4. *Determining a good metric for the fluency score is not trivial. I presented several candidate metrics. A possible metric is the speed of the translation process (disregarding correctness) by simply looking at the quantity of words in the source being translated per unit of time. This metric is not perfect, and I explained why I deem it plausible that a perfect a priori defined metrics might not exist. A metric can often (if not always) be improved by investigating the semantics of the (type of) natural language fragments that are being translated. Nevertheless, one could use one default metric that performs reasonably well, as long as a better one is not (yet) available.* ■

“

Research question 2.2.8 (accelerate SWiFT-Focused instance design). *SWiFT-Focused requires the development of a great many instances to be developed to cover most aspects of formal-fluency. How to design an approach in which this development can take place in an accelerated fashion?*

(Quoted from page 262.)

”

Conclusion 1 of research question 2.2.8. *As a first step toward at least streamlining the development process, a design template was presented (see section III.8.4 (p. 262)). This template indeed helped during development of the SWiFT-Focused variants that I designed. Moreover, I do not see how to do without such a template when working in teams of SWiFT-Focused-game developers.* ■

III.8.8.2 Future work

III.8.8.2.1 SWiFT-Focused

Future work on SWiFT-Focused could include the following.

- Extend the set of SWiFT-Focused challenges such that it covers an important part of formal-fluency. To support this process, organise an online open source community of developers and players of the SWiFT-Focused games.
- Create a ‘reasoner glue language’. To simplify the development of new SWiFT-Focused games, it would be nice to create a meta-reasoner-definition-language to more easily stitch existing reasoners together. Complementary to this, it would be nice if a compiler or interpreter would become available that also actually executes programs in this language *without* the necessity for the user of the language to have to install all reasoners, with their peculiarities, manually. It should automatically spawn or use existing instances of the reasoners being defined in the meta-reasoner language.

Chapter III.9

Conclusion and future work

III.9.1 Conclusion

The research questions posed in the introduction of this book part (section III.2.5 (p. 142)) and a summary of the corresponding research results achieved in this book part are as follows:

“

Research question 2.1 (survey fostering integration of automated-deduction). *What is a precise and operational definition of the systems that facilitate effective assistance of automated-deductive reasoners for people reasoning about their knowledge, and a survey of subchallenges and (already) existing solutions to create and foster the development of these systems?*

(Quoted from page 143.)

”

Conclusion 1 of research question 2.1. *The research based on reflection, literature study, and communication with experts such as Patel-Schneider (2015), that resulted in this part of the book showed that facilitating effective assistance by automated-deductive reasoners of people reasoning about their knowledge is a multi-faceted and complex challenge, along both the technological and the human dimensions (chapter III.3 (p. 151)). It turns out that even the development of a detailed and precise overview of the complete challenge itself is far from trivial. Therefore, one important conclusion is that instead of producing a static survey of the challenge, it is probably best to create a dynamic survey and a survey-development approach that can serve as a base for fur-*

ther refinement and adjustments by the research community. The current stage of the survey is to be found in chapter III.3 (p. 151). The most important aspect of the survey-approach is defining an overall operational challenge, that does not specify the solution (see definition 25 (p. 167) in chapter III.3 (p. 151)), however, that operationally defines the desired effects of the solution and the conditions the solution should meet. To the best of my knowledge, such an attempt has not been made before. Rather, we see a fragmented collection of research efforts in this domain (such as ‘the Semantic Web’, or ‘knowledge representation and reasoning’), where the overall goal is not or only vaguely defined, or consists of formulations that already hint at a solution. In doing so, they exclude alternative paths, and moreover, force themselves to give an overview of the complete solution, while it is probably impossible to reliably create such an overview, at least at present, given the aforementioned complexity of the challenge. ■

“

Research question 2.2 (fostering formal-fluency on a meta-level). *What meta level specialised extensions of the Orcoba-Approach exist that can contribute to fostering (the emergence of good composition-records for) formal-fluency?*

(Quoted from page 143.)

”

Conclusion 1 of research question 2.2. *First, all specialised extensions for the purpose of fostering formal-fluency are further extensions of general-Orcoba-extensions, including the most general of all: the Orcoba-Approach itself. These are systems that have not yet been specialised for a specific capability. Therefore, arguments about the effectiveness of these general systems to foster capabilities, also transfer to their successful capability-specific-Orcoba-extensions if these exist. The latter, of course, includes effective formal-fluency-specific extensions. These arguments have been the topic of section II.6.1 (p. 125) and in particular the conclusion 1 of research question 1.1 (p. 126).*

Second, the following arguments can be added concerning the existence of effective formal-fluency-specific extensions. First I discuss the compatibility of the Orcoba-Approach with the quest to foster formal-fluency, then I focus on concrete formal-fluency-specific Orcoba extensions.

Compatibility:

I believe the most general, unspecialised Orcoba has properties that make it suitable for being applied to fostering formal-fluency. Most notably formal-fluency lends itself well for expressing the method to practice it in a written form, and therefore, can be captured in a composition-record, one of the prerequisites for the application of the

Orcoba-Approach. Experimentation with some extensions indeed showed that participants were able to use the written composition-records quite naturally, and were able to solve the challenge purely by using these written composition-records.

The Challenge-Based-Orcoba, and in particular Or-evohut are extensions of the Orcoba-Approach that have been designed to make fostering human capabilities much more efficient. However, applying these to a certain capability, requires the definition of a suitable challenge-set, and a corresponding challenge-assessment (see section II.4.5 (p. 57)), which may not be easy or feasible for all types of (collective) capabilities. In this work, I believe I succeeded in defining good challenge-sets and associated metrics for formal-fluency, which meet the conditions of the mentioned extensions. See the next topic, 'Extensions', for more conclusions about these.

Extensions: This part has presented and explored several formal-fluency-specific extensions: real-time-collective-formal-thinking-Orcobas, with one further specialisation System- α -for-real-time-collective-formal-thinking, and the SWiFT-class-of-games, with the following further specialisations (and variants): SWiFT-Full, SWiFT-Fixtal and higher-order-SWiFT (the latter of which includes SWiFT-Focused).

In the SWiFT-class-of-games, challenge-sets are defined in the form of a translation game from natural language to formal-language. The metric is based on the degree to which one can deduce answers to queries from the resulting translation. The latter, in combination with the more detailed design decisions that have been presented in this part, form an almost direct expression of formal-fluency, and therefore a plausible metric. Moreover, I believe the challenge-set to cover many aspects of practicing formal-fluency 'in the wild'. Among other things, the text is broken up into smaller fragments, which stimulates the players to translate the text such that it results in a greater automatically-deducible-fraction. On the other hand, an insight is that there is no one single perfect challenge-set definition, because of various trade-offs in the area of formal-fluency. However, by adjusting the challenge-set accordingly, it should be possible to emphasise different aspects of formal-fluency.

The best SWiFT-game for fostering formal-fluency with regard to the challenge-set is SWiFT-Full (see section III.6.5.1 (p. 220)). It offers a challenge-set that is very close to the pursued-capability – in most of its facets. However, SWiFT-Full has some major disadvantages: it confronts the player immediately with full-blown formal-fluency, which can be intimidating, it is quite human resource-intensive and game sessions typically take a long time to play, which will most likely be a big hurdle for many to become and remain a player. Moreover, the resulting deduction algorithms created to deduce answers to questions are difficult to compare, and players have to balance the difficult trade-off between expressivity and computability while selecting a suitable formal-target-language. This prompted research and development activities into SWiFT games that complement SWiFT-Full.

SWiFT-Fixtal copes with some of these problems by fixing the formal-target-language for all teams. In this way, the complexity-expressivity trade-off problem is not on the plate of the players anymore, and it becomes much more straightforward to compare the deduction-algorithms of adversaries. Moreover, the difficulty level of the game can now, at least for a certain aspect, be built up gradually by choosing a (succession of) suitable formal-target-language(s). An experiment was carried out with an instance of *SWiFT-Fixtal*, in which the structural semantic conditions of the formal-target-language consisted of *RDF+S*, one of the simplest *KR*-languages. The experiment's results include: (1) The formal-fluency of the players reached a promising level, considering that this was the first experiment with novice players, (2) The game provided strong incentives for participating: among other things the players mentioned the attractiveness of the collaborative aspect and the intellectual challenge, (3) The main counter-incentives mentioned were the long duration of the game and, interestingly, again the intellectual challenge. More and detailed conclusions about the experiment are to be found in section III.7.6 (p. 240).

SWiFT-Fixtal (see chapter III.7 (p. 225)) copes to a certain, but not a sufficient degree with all of the mentioned problems of *SWiFT-Full*. This spurred me to do research into more games to extend the *SWiFT*-class-of-games. This resulted in the design of higher-order-*SWiFT*, an extension of higher-order-*Orcobas*. In higher-order-*SWiFT*, the full challenge of mastering formal-fluency is broken up into smaller subchallenges. This is an immediate solution to the first problem. Moreover, if the break-up meets some additional conditions it also allows an adequate reduction of human resources and playing time. For more detailed conclusions about this topic, see conclusion 2 of research question 2.2.3 (p. 298), conclusion 1 of research question 2.2.4 (p. 298) and the conclusion 1 of research question 2.2.6 (p. 299) starting on page 298.

The comprehensive nature of a composition-record for formal-fluency posed a problem for the application of *Or-evohut* for fostering formal-fluency. The problem is that minor changes to a large composition-record require a very large sample size of players using that composition-record to show their effect on the score. This problem was already addressed by higher-order-*Orcobas*, presented in section II.4.6 (p. 61). I believe that the design of higher-order-*SWiFT*, an extension of the latter, provides an adequate solution to this problem.

In retrospect, it turned out that the *Orcoba* extensions for real-time-collective-formal-thinking are less suitable for focusing on fostering formal-fluency. That is not a problem in the *Orcoba-Approach* as such, but a consequence of the fact that real-time-collective-formal-thinking is a more general capability than formal-fluency. That is why the *SWiFT*-class-of-games was developed: the *SWiFT*-class-of-games isolates formal-fluency. Real-time-collective-formal-thinking fosters another, more encompassing capability of which formal-fluency is a sub-capability. Nevertheless, real-time-collective-formal-thinking is worth pursuing both in the light of fostering collective-formal-thinking

(chapter III.5 (p. 197)) and formal-fluency. ■

“

Research question 2.3 (fostering formal-fluency on an object-level).
What are (ingredients of) composition-records that contribute to increased formal-fluency?

”

(Quoted from page 143.)

Conclusion 2 of research question 2.3. See section IV.2.4 (p. 320) and section III.7.6 (p. 240). ■

III.9.2 Future work

III.9.2.1 SWiFT

Future work concerning SWiFT in general may include the following.

1. Integrate evolutionary inspired extensions of the Orcoba-Approach into SWiFT. See chapter II.5 (p. 65) for more information about these extensions.
2. Specifically, integrate the evolutionary inspired extension Or-evohut into, preferably, all variants of SWiFT.
3. This book has presented SWiFT-Full, SWiFT-Fixtal and SWiFT-Focused. If needed, develop additional variants and versions of the SWiFT-class-of-games to increase the effectiveness of fostering full mastery of formal-fluency.

Future work concerning SWiFT-Fixtal and SWiFT-Full may include:

1. Experiment with smaller texts and bigger teams than in SWiFT-Fixtal- α , so that the text size per player is reduced, while the collaborative aspect is increased. See conclusion 1 of research question 2.2.1 (p. 240) for more details.
2. Develop an adequate solution for the algorithmic-defence-round. In the current designs of SWiFT-Full and SWiFT-Fixtal, the team has to carry out the algorithmic-defence-round. This may not be the best solution, because constructing the deduction-algorithms involves quite some additional training. Moreover, a difference in the skill of teams to construct such algorithms, may lead to an unfair score. After all, it is the quality of the translation that has be

determined, and not the capability of a team to construct deduction-algorithms. (Note that in the experiment with SWiFT-Fixtal- α , this problem was circumvented by letting the experimenter carry out the algorithmic-defence-round. However, this is not be a very scalable solution.) A solution could be to design another game that complements SWiFT, in which *other* teams have to create the best deduction-algorithms for a given translation.

3. Design an objective procedure to determine the validity of the constructed deduction-algorithms. Among other things, no other information may be used than information provided in the question (also see section III.7.4 (p. 235)). A centralised panel of independent reviewers is not likely to be very scalable. After all, it would be difficult for the panel to keep up with reviewing work if many people would be playing SWiFT. A solution could be to create a reviewing system that is based on peer review. Only if the competing teams unanimously agree with an algorithm, is it considered to be valid. Only in the hopefully rare event there is an unresolvable conflict, an independent arbiter is consulted.
4. Develop a solution for the text-assignment stage. In SWiFT-Fixtal- α this stage has been carried out by the experimenter in the role of game-coordinator. This is not a scalable solution. Directions for solutions include:
 - (a) Create a complementary game that organises a crowd that creates an ever-growing repository of texts.
 - (b) Develop software to automatically carry out the text-assignment stage. This may not be so easy, because the fragments should best be chosen such that they coincide with a fragment that has some internal coherence.
5. Develop a SWiFT-Full game to test and train full mastery of formal-fluency. In particular, solve the problem of how to compare the results of teams that work with different formal-target-languages and different automated-deductive-reasoners. See section III.6.4.2.1 (p. 219) for more details.

More detailed future work regarding SWiFT can, among other things, be found in: conclusion 1 of research question 2.2.1 (p. 240).

III.9.2.2 Transforming Wikipedia

SWiFT could be used to transform the content of a given large aggregate of natural language texts into a formal-language. In my opinion, a notable candidate would be Wikipedia, because of its comprehensiveness, uniform structure and relatively high quality. In the following Wikipedia is used as an example. However, it holds for

any aggregate of natural language texts, such as books, webpages, scientific articles, and so forth. In this setting, the SWiFT-class-of-games plays a double role. (1) It trains people in formal-fluency. (2) The result of their playing leads to a translation of Wikipedia into a formal-language with a high automatically-deducible-fraction. (*End of list.*) This enables people to consult Wikipedia using automated-deductive-reasoners. In other words, SWiFT could become a way to gamify and crowd-source a high quality transformation of Wikipedia into a formal-language. I think this is a highly fascinating and promising project.

For this purpose, the SWiFT game-designs presented in this book should be extended with a few other members or variants. Among other things:

1. The current SWiFT designs focus only on the degree to which automated-deductive-reasoners can be effectively applied to deduce information for the text that is used in a given translation-session. I developed the notion automatically-deducible-fraction in section III.3.5.2 (p. 174) to quantify essential aspects of this degree. However, in the scope of transforming Wikipedia, the automatically-deducible-fraction should be, ideally, optimised for the whole of Wikipedia. This is a challenging problem.

In this it is important to be aware of the following. Let t_1 and t_2 be two texts, and f_1 and f_2 two respective translations into a formal-language. Also assume that each translation realises an optimal automatically-deducible-fraction. These assumptions, unfortunately, do not imply that the union of the translations, $f_1 \cup f_2$, realises an optimal automatically-deducible-fraction with respect to the union of the original texts, $t_1 \cup t_2$. For example, f_1 could be using different names for constants that refer to the same entity. Hence, the union of translations of Wikipedia text produced in SWiFT-translation-sessions do not necessarily form a translation of the whole of Wikipedia that realises a high automatically-deducible-fraction.

Directions for solutions include: (1) Allow each team to use already translated texts in their algorithmic-defence-round in addition to their own translation. (2) In the text-assignment stage, choose a new text from Wikipedia, so one that has not been translated. Also, choose an old text that already has been translated. Also, select the best performing translation of the old text. During the translation-round the players translate the new text, and improve the existing translation of the old text. During the question-attack-round, the players attack the combined new and old text. This provides the incentive to optimise the automatically-deducible-fraction of the combination, and not only the new text. Because the old text already has been translated, though, the playing time will, hopefully, not be increased too much. Whether this approach is scaleable requires further research. (*End of list.*)

2. If you want to balance the trade-off between reaching the maximal automatically-deducible-fraction, computability, and translation effort, I think that allowing different formal-target-languages is essential. This is indeed allowed in SWiFT-Full. However, SWiFT-Full has not yet been fully implemented. In particular, it is not yet clear how to effectively cope with multiple formal-target-languages. How can you facilitate the algorithmic-defence-round across KR-bases expressed in different formal-languages? This probably requires the development of new techniques in addition to existing branches of research in this area.

This way, the SWiFT-class-of-games could also become a ‘playground’ for testing many aspects of the integration-of-automated-deduction into society (see chapter III.3 (p. 151)), and for associated research communities within fields such as knowledge representation and reasoning and automated theorem proving.

III.9.2.3 Real-time-collective-formal-thinking

A summary of a selection of future work mentioned in section III.5.6.1 (p. 208) and section III.5.7 (p. 210) is as follows. Although SWiFT has been a response to solve some major problems encountered during experimentation with real-time-collective-formal-thinking, it cannot replace it. This has been explained in section III.5.6.1 (p. 208). Therefore, it is important to do future R&D into real-time-collective-formal-thinking to bring it beyond the first system implemented to support it, System-Real- α . This R&D may include the following.

1. Facilitate the coordination of the collective thinking process. For more details see section III.5.7.4 (p. 212).
2. Integrate Or-evohut into System-Real- α . This is not easy, because it is hard to define a good scoring procedure. This has been explained in section III.5.7.3 (p. 211).
3. Facilitate an effective decentralised development of educational materials to educate participants about the information expressed in a given EERG. The more people understand the content of an EERG, the more people can contribute to growing collective insight into matters. Section III.5.7.5 (p. 213) provides more details.

Part IV

Experimentation

Chapter IV.1

Overview and structure

This part presents experiments with some of the systems developed in the earlier parts. The main purpose is to validate design decisions in the light of their envisioned purpose, and collect insights for improvement of these systems. The structure of this part is quite simple: each chapter contains an experiment, and can be read in isolation. In other words: the chapters do not build upon each other concerning reading-order. Relations between the experiments are implicit: they can be found in the chapters in the previous parts that define the systems, and, among other things, mention the experiments as source of design decisions. In general, the chapters are presented in the order in which the associated systems are treated in the previous parts.

Note that the experimentation in this part focuses on formal-fluency. Full experimentation with the Orcoba-Approach and in particular with Or-evohut- α is future work, as has been motivated in section II.5.9.2. Nevertheless, aspects of the Orcoba-Approach have been covered in the experiments with SWiFT.

Chapter IV.2

System·Real- α

IV.2.1 Introduction

This chapter contains a report of experimentation with System·Real- α , or in full: System- α -for-real-time-collective-formal-thinking. This system has been described in chapter III.5 (p. 197).

For the setting during the experiment a few things have to be added to System·Real- α . The participants stay in contact with the experimenter through a tool for audio conferencing. The experimenter follows what happens from his own work station, without participating in the session. The participants and the experimenter may request to suspend the session temporarily to discuss aspects of the methods which do not appear to be clear. The audio tool is used exclusively during the suspensions, for the rest there is silence. Moreover, during the suspension the participants may not exchange information about how to solve the problem. In the sessions all participants worked from their own working places or homes, always in different rooms, and sometimes in different buildings, well outside of visual or auditory range from each other.

IV.2.2 Goal

The two concrete goals of this chapter are the following: (1) to evaluate the quality of entity definitions produced by participants and the degree to which the participants approached translating their thoughts in real-time and (2) to suggest how to improve the future performance of the participants.

This experiment is limited to the four properties to which entity definitions should

comply mentioned in section III.5.5 (p. 206): (1) being unambiguous (<**Unambiguous**>); (2) having an extensional meaning (<**ExtensionalMeaning**>); (3) having a generic description (<**GenericDescription**>); (4) defining a single entity (<**SingleEntity**>).

I assume that all four contribute to a significant increase in the degree to which automated-deductive-reasoners can be applied effectively. They have been further explained and motivated in section III.5.5 (p. 206). However, the main focus of this chapter does not lie on validating whether these properties lead to an increased effectiveness of automated-deductive-reasoners, but lies on evaluating to what extent the participants produced results in agreement with these properties. In this chapter the validation of effectiveness is exclusively based on argumentation, not on experimentation, a topic which, however, is in the focus of SWiFT.

My concluding suggestions to improve participants' future performance based on this experiment can be classified into suggestions to: (1) change the essential part of the method described in the composition-record itself, (2) increase the probability that participants work in accordance with this method, without changing it essentially (for example: by integrating approaches that prevent participants from forgetting to apply parts of the method) and (3) refine and adjust the focus of study, in order to invest my resources in the aspects I deem most important in this phase of the study.

IV.2.3 Results of experiments

For easier reference the experiment sessions and the participants are labelled in the following way:

- <**RaZo**>: Rabin and Zoë
- <**SaMi**>: Sam and Mitch
- <**MaJo**>: Marc and Joe
- <**LaHo**>: Larissa and Howard

Each team chose one of the following problems:

<**problem concatenation**> When you concatenate the decimal representation of the numbers 1 to 12, you get the number 123456789101112, which contains 15 digits. When you do the same for the decimal representation of the numbers 1 to n you get a number with a decimal representation that consists of 1788 digits. What is the value of n ?

<**problem add to 6**> How many positive numbers smaller than 1000 have a decimal representation in which the digits sum up to 6?

In sessions <**LaHo**>, <**SaMi**> the choice was: <**problem add to 6**>. In sessions <**RaZo**>, <**MaJo**> the choice was: <**problem concatenation**>.

Table IV.2.1: Average quality of real entities.

RaZo	SaMi	MaJo	LaHo	Session Name
90	90	60	105	Session Duration (min)
13	26	12	44	# ¹ Entities Created
9	17	12	25	# Entities per Hour
13	15	8	15	# Real Entities ² Created
9	10	8	9	# Real Entities per Hour
4	9	6	9	Grade Entities ³

IV.2.3.1 Quality of defined entities

Table IV.2.1 provides the average grade assigned to the entities created for each team. A detailed definition of the way points were assigned is omitted. It can be obtained by contacting the author. The total number of entities created is also provided, among other things to give the sample size on which the average mark was based.

Sometimes, the descriptions of entities are very similar. When a decision how to express one of them is taken, the other entities naturally follow from them. In such cases, these collections are counted as only one entity for determining the average quality. For example: the entities {"■ represents the integer 5."} and {"■ represents the integer 17."} are counted as one entity. Not doing this would give a distorted picture of the average performance of the team.

IV.2.3.2 Core properties

The following investigates to what extent the descriptions produced by the participants do agree with the required core properties (as formulated in section III.5.5 (p. 206)). For each property one typical nonconformity found in the results will be treated. If applicable, I have tried to reconstruct the thought pattern of the participant which gave rise to the nonconformity. This analysis is the basis for the suggestions for improving the method (composition-record) in section IV.2.4 (p. 320).

In some cases, an alternative modelling will be given that agrees with the method. When inescapable, it will be a multi-entity model.

Note that the following uses the preferred label of each entity as a label to refer to the entity which was defined by the participant. Before starting reading this section, the reader is suggested to look at the notation for EERGs in section III.5.4.1.1 (p. 203).

¹"#" translates as: "number of".

²real entity: counting entities that have a description that is almost identical as one, 'real', entity.

³On a scale from 0 (bad) to 10 (excellent).

For the sake of brevity, the human descriptions of the entities are sometimes omitted. In the complete EERG these descriptions are, of course, included.

IV.2.3.2.1 <SingleEntity>: {SolutionDescription}

One of the cases in which the <SingleEntity> criterion was not met was the following:

{23}—{hasHumDescr}→{“ N when $N < 10$, $N + (N - 9)$ when $N < 100$, $N + (N - 9) + (N - 99)$ when $N < 1000$, and so on.”}

An alternative definition more in agreement with the method is:

{23}—{hasHumDescr}→{“■ represents the function that assigns to integer N : N when $N < 10$, $N + (N - 9)$ when $N < 100$, $N + (N - 9) + (N - 99)$ when $N < 1000$, and so on.”}

Reconstruction of the thought pattern of the participant: most probably he followed the question-reply pattern. This tempted him to formulate a description that only makes sense after reading the description of the problem (see also section IV.2.4.3 (p. 321). However, each entity should have an existence on its own, and the description should be sufficient to define it unambiguously.

IV.2.3.2.2 <GenericDescription> and <ExtensionalMeaning>: {“maximum value...”}

Observe the following description: {93}—{hasHumDescr}→{“maximum value of formula for digit count of Orcoba experiment 5 where numbers are of two digits.”}

The extension of this description is an integer, thus this is not the most generic way to describe that integer. Moreover, the description of the function mentioned in the description is quite ambiguous. An alternative definition that is more in agreement with the method is:

- {139}—{hasHumDescr}→{“■ represents the function $f(n) = [\text{total number of digits in the concatenation of the decimal representations of the numbers } 10 \text{ to (and including) } n.]$.”}
- {151}—{hasHumDescr}→{“ x —■→ y expresses that y is the maximum value of the mathematical function x .”}
- {139}—{151}→{100}

{100} is the entity which the participant was looking for, an integer which complies to certain constraints. Compare this with the number of inhabitants of Japan example given in section III.5.5 (p. 206).

IV.2.3.2.3 <Unambiguous>: {Joe's first rough thought ...}

Observe:

- {56}—{hasPrefLab}→{"Joe's first rough thought about the value of n ."}
- {56}—{hasHumDescr}→{"99 < n < 999"}

Analysis of thought pattern: n in the description of {56} refers to the variable occurring in the problem description. This makes the description highly ambiguous, as it cannot stand on its own. Its meaning can only be disambiguated when you know this problem description, and moreover it only makes sense in that context. The suggested improvement can be found in section IV.2.3.2.4, together with an analysis of the label.

IV.2.3.2.4 <LabelRequirements>: {"■ represents Joe's first rough thought ..."}

The label of {56} in section IV.2.3.2.3 contains additional information the participant intended to express. Most probably he wanted to do a few things: (1) state a range of numbers (2) state that the solution is in that range (3) state his current opinion that the probability is high that latter is true. In accordance with the method would have been:

- {674}—{hasHumDescr}→{"■ represents the mathematical set $\{\{1\}, \{1, 2\}, \{1, 2, 3\}, \{1, 2, 3, 4\}, \dots\}$ "}
- {17}—{isInstanceOf}→{674}
- {17}—{hasTotalNumberOfDigitsInDecimalRepresentation}→{234}
- {234}—{hasHumDescr}→{"■ represents the integer with decimal representation 1788."}
- {100}—{isTheGreatestNumberIn}→{17}
- {100}—{isElementOf}→{SetOfNaturalNumbers} (Note: {100} is not the integer 100, but an identifier for such an entity, as explained in section III.5.4.1.1 (p. 203).)
- {101} = {100}—{isElementOf}→{NaturalNumbersFrom99To999}
- {JoeTynerBorn22June1982inVancouverCanada}—{intuitively judges to be true}→{101}

Moreover, in this way both the concept of 'thought' and 'problem' are avoided. In section IV.2.4.4 (p. 322) I will defend why I think this might be a more fruitful approach.

IV.2.4 Conclusion

IV.2.4.1 Performance

The performance consists of two aspects: the quality of the entity definitions and the speed with which they were created. In general, the degree to which the entities agreed with the core properties was quite high, in spite of the limited training of the participants. Two teams scored an average of 9, and the other two of 4 and of 6 (on a scale of 0 (bad) to 10 (excellent)). All teams created about 10 real entities per hour. I assume that this number is still far removed from a fluent translation of thoughts. However, among other things, we have to keep into account the extra time needed to define the entities and the fact that the version of the software used during the experiment turned out to be too slow. Therefore, I conclude that the method is feasible, and future performance could be improved significantly after refinement of the method and more training.

Moreover, the difference in scores corroborates that there are considerable differences among individuals in applying the method, which supports my objective to customise the training and the method.

IV.2.4.2 Reducing human fallacies

Suggestions to reduce the probability that human fallacies will occur were already covered in section III.5.6.2 (p. 209). The following repeats them, but adds the observed nonconformities by which it was inspired between square brackets.

1. (a) The participant should pay extra attention to **<SingleEntity>** when a parameter occurs in a human description (such as for example x , N , q , etc.). [{23} in section IV.2.3.2.1 (p. 318)]
 - (b) The software scans for symbol sequences in the human description that could be parameters (this can be checked quite easily on a syntactical level) and warn the participant to double check **<SingleEntity>**. [{*SolutionDescription*} section IV.2.3.2.1 (p. 318)]
2. The participants are made aware of the following pitfall. When solving a problem, one's thoughts often unfold in sequence of (sub)questions and (sub)responses (or (sub)problems and (sub)solutions). A pitfall in this process is defining entities that form the 'response' to a question such that you need to read the entities that form the 'question' to understand them. For example, given problem context "the number of orders in which N different objects can be placed". With this in the back of one's mind, one might easily be tempted to define the 'entity' {"■ represents the number N !."}, which is not a

<SingleEntity>. [{{SolutionDescription}} section IV.2.3.2.1 (p. 318) {Joe's first rough thought ...} section IV.2.3.2.3 (p. 319)]

3. (a) The software first lets the participant define the human description, and after that gives the option to attach a label, too. (Currently, the order is exactly the opposite, and on top of that, a label is obliged).
- (b) The software only allows an entity description to be submitted for judgement after the participant has run through a check-list based on the properties to be pursued.
- (c) Only after a description has been accepted by all participants, a participant may optionally create a label. [{{SolutionDescription}} section IV.2.3.2.1 (p. 318); {Joe's first rough thought ...} section IV.2.3.2.4 (p. 319)]

IV.2.4.3 Changes to the essence of the method

Redefinitions fundamentally changing the method were already mentioned in section III.5.6.2 (p. 209). The following repeats them, however, it adds the nonconformities that inspired them between square brackets:

1. Problems may not be defined as entities. In fact, an information-related problem can in most (if not all) cases be decomposed into two elements: incomplete information and an intention to complete that information. The second makes the first a "problem". Transparency would be increased when these two elements are separated explicitly. *Example: Instead of posing the problem "Who is the writer of Tao Te Ching?", which is both a set of incomplete information as an intention indicated by the question form, it can be reformulated as intentionless incomplete information: "The writer of the Tao Te Ching is ...". In EERG form this would be: {The Book Tao Te Ching}—{was written by}→{77}, in which {77} represents the writer of the book, unknown as yet.* The participant can express the intention of it being a problem by indicating the pieces of information he would like to see completed, by adding nodes and relations that express this. In the example, such a node and relation would be added to the entity {77}. This contributes to reusability at least in the fact that the formulation of answer and reply are one and the same. [{Joe's first rough thought ...} section IV.2.3.2.4 (p. 319)]
2. Defining labels is optional. In some cases, a label is not necessary. Forcing participants to give a label anyway, could lead to several undesirable situations, including: labels being (almost) equal to the description, which is superfluous, or worse, labels containing additional information which should have been

expressed with additional entities. [*Joe's first rough thought ...*] section IV.2.3.2.4 (p. 319); SolutionDescription section IV.2.3.2.1 (p. 318)]

IV.2.4.4 Changing focus

Perhaps it is better to exclude the possibility to express the content of thoughts as separate natural text entities in an EERG. Allowing this creates a shortcut around the core of the method: expressing thoughts in a reusable way. The same can be said for any “information entity” (such as documents and audio files). Each of them can contain information that should have been expressed as a combination of entities and relations with an elementary meaning.

However, note that I deem it important to finally include this option to allow participants to choose themselves where they want to invest energy of creating high quality knowledge representations, or for example incrementally improve the representations of their knowledge. The main motive for the narrowing of focus is that I want to give priority to the aspect of exchangeability of information.

IV.2.5 Future experimentation

A problem in drawing conclusions from the experiment is its low sample size: in total 95 entities and 53 real entities created. This problem is easily solved by scheduling further experiments.

Chapter IV.3

SWiFT-Fixtal- α

IV.3.1 Introduction

This chapter contains a report of experimentation with SWiFT-Fixtal- α . This system has been described in chapter III.7 (p. 225).

IV.3.1.1 Game setting during experiment

The following briefly repeats the game phases as described in section III.7.4.1 (p. 235), and supplements them with information about the experimental setting. Text preceded by ‘*Session*’, is taken from the actual experiment.

Training. The players thoroughly study the rules of the game and the composition-record.

Session. In the session described in this chapter the players studied the material (which included a complete tutorial of SWiFT-Turtle and the guidelines node definitions) and took a multiple choice test. This took them about 90 minutes.

Text assignment. The coordinator divides a text into as many non-overlapping fragments as there are players in a team. Each player of each team then receives a fragment that is different from that of his peers. Consequently, each team covers the complete text. Players do not see the fragments of their team members.

Session. In the session the players were confronted with an encyclopaedic text

explaining the concept *vitamin*. The text consisted of 550 words. The article was divided into three fragments, the first introducing the concept, the second describing its history and the third treating the relation between vitamins and health. For example, the following sentences form part of the second fragment:

“In 1931, Albert Szent-Györgyi and a fellow researcher Joseph Svirebely discovered the chemical structure of vitamin C: ‘hexuronic acid’. In 1937, Szent-Györgyi was awarded the Nobel Prize for his discovery. In 1943 Edward Adelbert Doisy and Henrik Dam were awarded the Nobel Prize for their discovery of vitamin K and its chemical structure.”

And an example from fragment 3 is the following:

“A source of vitamin K is Alfalfa.”¹

Translation round. The teams translate the text into a SWiFT-Turtle representation. They follow the guidelines prescribed by the composition-record as much as possible, including the consensus strategy.

Session. Team 1 translated the previously given examples as follows:

```
<"Edward Adelbert Doisy"> <type> <"theConceptOfNobelPrizeWinners">.
<"Henrik Dam"> <type> <"theConceptOfNobelPrizeWinners">.
<"Szent-Györgyi"> <type> <"theConceptOfNobelPrizeWinner">.
<"The concept of vitamin K"> <"hasSource"> <"The concept of Alfalfa">.
```

Question-attack-round. The teams challenge each other’s translations by posing questions that have a specific answer based on the content of the text. *Session.* One of the questions produced was: “What is a source of the vitamin that two researchers got the Nobel Prize for?”. The expected answer, based on the text is “Alfalfa”.

Algorithmic-defence-round. Each team then attempts to construct an algorithm that deduces the correct answer from their translation. The algorithm should not contain other information than the information provided in the question. In the present version of the game, the language to construct the algorithms is SPARQL, consequently the algorithms take the form of SPARQL queries. In the experiment, the algorithmic-defence-round was carried out by the experimenter (the author). If

¹This information was actually provided in a table in the text.

the players would have had to do this themselves, it would have taken much more time to prepare the players for playing the game.

Session. The following was a query constructed for team 2 to deduce the answer to the above question:

```
select distinct ?s where
{
  ?n1 rdf:type <theClassOfNobelPrizeWinners>.
  ?n2 rdf:type <theClassOfNobelPrizeWinners>.
  ?d rdf:type <Discovery>.
  ?d <wasDoneBy> ?n1.
  ?d <wasDoneBy> ?n2.
  ?d <discoveredMaterial> ?m.
  ?m <hasSource> ?s.
  filter(?n1 != ?n2).
}
```

Its result was:

```
Result 1. <The_concept_of_Alfalfa>
```

Note that the following query, although it produces the correct answer, is not allowed because it contains information that is not provided in the question:

```
select ?s where
{
  <The_concept_of_vitamin_K> <hasSource> ?s.
}
```

The fact that the Nobel Prizes were won for the discovery of vitamin K is in the text, but not in the original question.

Determining the score. *Session.* See section IV.3.2.1 on the next page.

IV.3.1.2 Additional elements of the experimental setting

Additional details concerning the experimental setting are as follows. The players stay in contact with the experimenter through a tool for audio conferencing. The experimenter follows what happens from his own work station, without participating in the game. The participants and the experimenter may request to suspend the session temporarily to clarify game rules that are not yet clear. This time is subtracted from the playing time. All players worked from their own working places or homes, always in different rooms, and mostly in different buildings, outside of visual or auditory range.

IV.3.2 Results and analysis

This section presents the results and analysis of the experiment. These are used in several other sections throughout this document to answer associated research questions. Section IV.3.2.1 opens with the score of the teams with regard to formal-fluency. Subsequently, to verify whether the questions produced during the question-attack-round formed a good means to assess the quality of the translations, I briefly reflect on them in section IV.3.2.2. Section IV.3.2.3 on the facing page presents observed violations of the selection of composition-record that have been presented in section III.7.3 (p. 227). Section IV.3.2.4 (p. 330) concludes with observations related to motivation.

IV.3.2.1 Score

With regard to the algorithmic-defence-round, team 2 won decisively against team 1. In total 19 legitimate questions were used in this round. Team 2 scored 10.3 points, 1.8 times better than team 1 with 5.7 points, using the score function defined in section III.7.4.1 (p. 235). More data shows that this does not contradict intuition. First, team 2 correctly answered in total 12 questions, against 8 for team 1 (also see fig. IV.3.1 on the facing page). Second, each question both teams answered correctly (in total 4), the complexity of the query of team 2 was lower.

In contrast, the total playing time for team 1, 209 minutes per player, was shorter than that of team 2, who played 238 minutes.

The overall score for team 1 is 27.3 and for team 2 it is 43.3, which means team 2 scored 1.6 times better than team 1. In other words, according to the scoring formula, team 2 made the translation most fluently.

IV.3.2.2 Question-attack-round

During the question-attack-round, the players produced 19 legitimate questions. These questions included for example: “What is a compound that can act as vitamin C?”, “Where can vitamin D be synthesized?” and “What is a biological function of the vitamin that was found the earliest?”.

Indications that this set of questions is suitable to assess the quality of the translations are the following: (1) Quite some questions appeared to require information from more than one fragment to be answered. (2) The information required to answer the questions was spread out over the text. (3) The answer to most questions was not represented directly in the text, but required deductions to be made.

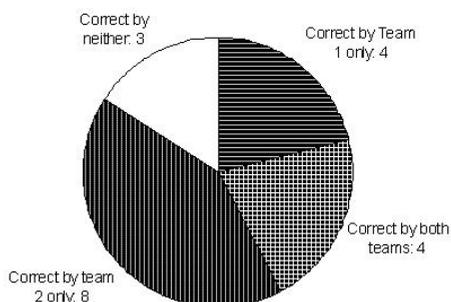


Figure IV.3.1: Performance Algorithmic Defence

IV.3.2.3 Violations guidelines node definitions

This section treats a selection of violations present in the node definitions created by the players. Moreover, it presents an overview of the suspected causes of the violations, including reference to related literature when applicable. It elaborates on the suspected causes for these violations by (1) presenting an exemplary fragment from the translation, (2) explaining why it contains a violation, (3) generalising the fragment so that it can be used to prevent as many errors as possible, (4) attempting to reconstruct the thought pattern that led to this violation, (5) investigating whether the violation led or could lead to queries that are more complex (or no possibility to construct a query at all) which would form an indication that the associated conditions should indeed be part of an effective composition-record.

IV.3.2.3.1 Violation of article 1 (p. 232): The definition of each node must form a fixed reference to an entity

Cause 1: Deceiving natural language subjects

Example: <<"■ represents the concept some animals.">>.

Explanation: see article 1 (p. 232).

Thought pattern: The expression "some animals" can be used in natural language sentences in the same way as for example "the hippopotamus". E.g. "Some animals are big", and "The hippopotamus is big" have the same grammatical structure. Both expressions occupy the role of natural language subject, in spite of the fact that the subject of the latter forms a fixed reference to an entity (namely the concept

hippopotamus), while the subject of the first does not. This makes it very tempting to make the mistake to define “some animals” as a node.

Generalisation: in natural language, that what are actually parameters (place holders) in sentences often take the appearance of specific occurrences, while they are not. Examples include text-fragments that contain the notions “some”, “all”, “a few”, “nothing” etc.

Effects on query: No effects during this session, but could have had as explained in article 1 (p. 232).

IV.3.2.3.2 Violation of article 3 (p. 233): Definitions must be univocal

Cause 1: Lack of awareness of the metaclass/superclass ambiguity

Example: <<"■ represents the concept of vitamins.">>

Explanation: The following two triples show that this definition is ambiguous:

```
<"The concept vitamin A"> <subclassOf> <"The concept of vitamins">.
```

```
<"The concept vitamin A"> <type> <"The concept of vitamins">.
```

(Both triples were encountered in the translation of a team.) In the first case, the definition is interpreted as a concept that is a *superclass* of the different vitamins (A, B, etc.). In the second case, the description is interpreted as a concept with the different vitamins as instances. Given the fact that each particular vitamin is also a class, in the last case the concept forms a *metaclass* (a class of classes) (Schreiber, 2002).

Thought pattern: The metaclass/superclass ambiguity is present in the natural language word “vitamin”. The players were not aware of this type of ambiguity, because when using natural language for normal human communication, they disambiguate this unconsciously.

Effect on query: Yes, the query to answer one of the questions became longer than needed for team 2 (a SPARQL UNION-operator was required).

Cause 2: Lack of awareness of ambiguity in general

Example:

```
<<"■ represents the relation that is defined as follows:  $x \dashrightarrow y$  if and only if  $x$  is a Vitamin and  $y$  is the place where  $x$  was discovered.">>
```

Explanation: It is not clear whether “place” means the geographical location where the discoverer made his discovery, or the material in which the vitamin was first encountered.

Thought pattern: The players didn’t realise the ambiguity of the word “place”.

Generalisation Trivial generalisation (which means that it will hold for most violations of the univocality condition): players are not aware of the ambiguity in their definition.

Effects on query: No effects during this session, but could have led to faulty deductions if the node would have been interpreted in another way to build a triple.

IV.3.2.3.3 Violation of article 4 (p. 234): Avoid extension/intension ambiguities in definitions

Violations of article 4 (p. 234) found:

Cause 1: Reference by property

Example: <<"■ represents the concept of ultraviolet wavelength.">>

Explanation: This description is ambiguous in that it is not indicated whether the extension (a specific distance) or the intension of the description (the property of being ultraviolet wavelength) is intended.

Thought pattern: The text reads: "...with the help of *the natural ultraviolet wavelength of sunlight...*" (emphasis added). The participant mimicked that construction. It is interesting to note that this reveals another form of ambiguity, next to the intension/extension explained in 'Generalisation'.

Generalisation: This error is likely to occur during translation of all natural fragments with the following ambiguity: the fragment is intended to refer to an object with a certain property (in the above case: *the part of sunlight* with an ultraviolet wavelength), while it is phrased in such a way that we would normally interpret it as referring to the property itself (the ultraviolet wavelength, which is nothing but a distance). However, the fragment can (at least) be interpreted in these two ways.

Effects on query: No effects during session, but could have had as explained in article 4 (p. 234).

IV.3.2.3.4 Violation of article 5 (p. 234): Definitions must be unique references to entities

Cause: Expressing different names of the same entity.

Example:

<<"■ represents the relation that is defined as follows: $x \dashrightarrow y$ if and only if x is equal to y .">>

<"The concept of vitamin B7"> <"isEqualTo"> <"The concept of Biotin">.

Explanation: The rule ‘Definitions must be unique references to entities’ (article 5 (p. 234)) implies that one may not use the relation <"isEqualTo">. If you would need it, you should retract one node, because, apparently, it has been defined two times. Moreover, the rule ‘Definitions must be unique references to entities’ (article 5 (p. 234)) in combination with the rule ‘Avoid extension/intension ambiguities in definitions (article 4 (p. 234))’ disqualify the second triple mentioned in the above example in yet another sense. The text-fragment implies that the word ‘Biotin’ has the same extension as the word ‘vitamin B7’. According to the rule ‘Avoid extension/intension ambiguities in definitions’, the corresponding nodes, therefore, have an identical meaning. This is not allowed according to the rule ‘Definitions must be unique references to entities’.

Thought pattern: According to the player the text expressed that vitamin B7 and biotin are synonyms. Internally she rephrased this information in the natural language construction “Vitamin B7 is equal to biotin” before creating <"The concept of vitamin B7"> <"isEqualTo"> <"The concept of Biotin">..

Generalisation: Expressing information about different names of the same entity. Specifically, caution is required when dealing with natural language words such as “same as”, “equal to”, “identical to”, etc.

Effects on query: No effects during session.

IV.3.2.4 Motivation

During the experiment the following observations have been made that are relevant for investigating whether the game is a suitable environment to foster composition-records. These are for the most part verifiable because voice recordings were made.

Motivation.

Incentives. The players of several unrelated institutes voluntarily dedicated many hours in playing the game. Several players emphasised that the collaborative aspect was highly motivating, and confirmed that an even stronger emphasis on this aspect would have a positive influence on their motivation. Related to this was that some expressed that the fact they were curious about the quality of the end-product of their collaboration was motivating. One player expressed that the game, specifically the task of translating natural language texts, had a positive effect on her work in formal knowledge modelling. One player expressed that the intellectual challenge posed by the game made it much more attractive for her than simpler games for creation of formal representations. Another player expressed he would be prepared to continue playing the game about one hour a week, beyond the time reserved for the

experiment. No player, in spite of the long duration for an experiment, discontinued the game for other reasons than circumstances beyond his or her control. During a session, two players requested to continue playing even after the experimenter indicated he wanted to suspend it.

Counter-incentives. The time needed to participate in the game and its intellectual challenge made it more difficult to recruit players for the experiment. The size of the original text to translate caused some volunteers to complain, because after playing a while they noticed it would take them too much time (therefore, I reduced it to about 180 words per player).

Suggestions to improve the composition-record. Suggestions made by the players included the following: (1) introduction of syntax for more than binary relations, (2) improved mechanism for quickly finding back nodes of the other participants, (3) more intuitive interface for judging node definitions, (4) pre-printing of the required part of node definitions, (5) change to a more expressive language than RDF+S, because they realised some information could not be translated and (6) investigate how to deal with vagueness in natural language.

Part V

Conclusion and future Work

Chapter V.1

Conclusion

This book addressed two related central quests, one general and one specific, also expressed in the research questions posed in section I.1.3 (p. 7), which are repeated below including the concluding results.

“

Research question 1 (the general quest). *How to foster human·collective-intelligence (or briefly: hucolligence)?*

(Quoted from page 7.)

”

Overall Conclusion of research question 1. *This work is based on three principles for fostering Hucolligence that have been expressed, and motivated, in the general introduction of this book: decentralised·development, institutionalised·self-reflection and human-artefact-co-evolution (section I.1.4.1.2 (p. 9)). The Orcoba-Approach has been developed in chapter II.3 (p. 43) as a top-level implementation of these three principles. The Orcoba-Approach is based on the concept of the Orcoba, which is a community-pursuing-a-capability, together with a record (composition-record) that allows the composition of their capability to be (at least partially) reproduced and examined. Moreover, the Orcoba's composition-record is open to the public for reuse, and is designed or supported by all members of the Orcoba. Additionally, the composition-record contains records pertaining both to the artefacts and the human capabilities. This work explained why this offers a solid top-level implementation of the given principles. (Also see conclusion 1 of research question 1 (p. 125).)*

However, the Orcoba-Approach system(-class) satisfies the aforementioned basic

principles predominantly by ‘definition’ – to make a system that actually realises these principles and the overall goal to contribute to increased hucolligence, gives rise to several subchallenges that are non-trivial to solve. This gave rise to research question 1.1 (p. 48) about establishing a part of these subchallenges. Several effective (directions for) solutions in the form of Orcoba extensions have been presented and explored in this book.

Among other things, the amount of (human and other) resources to assess the quality of solutions is finite per time unit. How can one distribute these resources effectively over the provided solutions? This work presented, among other systems inspired by evolutionary theory and evolutionary computation, Or-evohut, a promising solution to this subchallenge. More detailed conclusions concerning the subchallenges and the proposed solutions are to be found in the conclusion 1 of research question 1.1 (p. 126). ■

“

Research question 2 (the specific quest). *How to foster human-fluency-in-formal-languages-for-automated-deduction (or briefly: formal-fluency)?*

”

(Quoted from page 7.)

Conclusion 1 of research question 2. *Formal-fluency is not a means to its own end. In this work, formal-fluency is considered part of the greater system of facilitating large-scale and effective assistance by computers of people reasoning over their knowledge. Understanding this broader context may be important for fostering formal-fluency, and moreover it may be more satisfactory to have an overview of the overarching research question of which fostering formal-fluency is a part. This gave rise to the sub-research-question to create such a survey (research question 2.1 (p. 143)). The resulting research confirmed that this is a multi-faceted and complex challenge, and that this even holds for specifying the challenge and its subchallenges. Therefore, this work presented an approach for developing a dynamic survey, that is, a survey that can be improved effectively. More detailed conclusions about this topic can be found in the conclusion 1 of research question 2.1 (p. 303).*

Within the greater system, this work focused on R&D of systems for fostering formal-fluency. As stated in the introduction of this work (section I.1.4.2 (p. 11)), it pursued these on two levels, the meta level and the object level.

On the meta-level, this work presented and explored several extensions of the Orcoba-Approach that are promising for the purpose of fostering formal-fluency. More detailed conclusions about these extensions are to be found in conclusion 1 of research question 1.1 (p. 126).

For detailed results regarding the object-level, please be referred to conclusion 2 of research question 2.3 (p. 307). ■

Chapter V.2

Future work

V.2.1 Fostering hucolligence with Orcobas

Future work in the field of fostering hucolligence with the Orcoba-Approach should first and foremost realise a completion of a software system that implements a basic version of the most advanced Orcoba-extension presented in this work, Or-evohut- α . With a full implementation of Or-evohut, the Orcoba-Approach can actually be applied to a plethora of human capabilities, both as a means to foster them, and as means to collect new data to develop improved extensions of the Orcoba-Approach. Moreover, giving priority to the development of higher-order-Orcobas is a good idea because I believe that the latter will play an essential role in the effectiveness of the Orcoba-Approach.

For more details see section II.6.2 (p. 126).

V.2.2 Fostering formal-fluency

V.2.2.1 SWiFT

A summary of a selection of future work mentioned in section III.9.2.1 (p. 307) is as follows.

1. Develop an implementation of SWiFT-Full. Within the SWiFT-class-of-games this is an indispensable instrument to foster full mastery of formal-fluency.
2. Find organisationally scalable solutions for certain rounds of SWiFT-Full and SWiFT-Fixtal so that they can be played by large crowds.

3. Integrate Or-evohut into, preferably, all variants of SWiFT.
4. Launch a project to use SWiFT as a means to crowdsource the transformation of Wikipedia, or any other aggregate of natural language texts, into a formal-language. See section III.9.2.2 (p. 308).

V.2.2.2 Real-time·collective·formal·thinking

A summary of a selection of future work mentioned in section III.9.2.3 (p. 310) related to real-time·collective·formal·thinking is as follows. The suggestions cover any aspect needed to enable real-time·collective·formal·thinking, and is not limited to formal·fluency only. (1) Develop personalised learning routes towards understanding of the content expressed in a given KR-base (see section III.5.7.5 (p. 213)). (2) Develop instruments to coordinate the collective thinking process (see section III.5.7.4 (p. 212)). (3) Develop instruments for version management (see section III.5.7.2 (p. 211)). (*End of list.*) More details and more suggestions for future work have been presented in section III.9.2.3 (p. 310) and section III.5.7 (p. 210).

Afterword

Several paths culminated in a project of which this book that lies in front of you forms a spin-off, some personal, some associated with my love and concern for the world around me. On closer inspection, these paths unfolded against a unified background. I first sketch the paths and then explain what they have in common. First a disclaimer: as a reader, you do not have to agree with all statements made in the following to fully appreciate the content of the book. The content of the book has a limited, mostly scientific and technological scope, and the validation of the instruments developed in it is independent of the statements made here. The following provides the broader context in which these instruments came into existence.

The path of the homo universalis. From childhood, I have been strongly attracted to the notion 'homo universalis' – the person who moves across a wide range of disciplines in depth and, in doing so, integrating these and serendipitously making discoveries that require the combination of several disciplines. I have always felt a profound resonance with essentially all disciplines and other dimensions of existence, ranging from sports, music, economics, biology, psychology, technology, mathematics, architecture, craftsmanship, art and interpersonal relations. I felt this for each individually, for their interconnections, and in what lies beyond them.

The path of learning. I have always learnt with devotion and have had a natural interest in the context and origin of the things I learnt. For me, learning is a journey. Just reading a book, or acquiring a skill, can bring you into contact with a borderless and beautiful world of insights and experiences, in which you soon find out that everything is somehow connected.

The path of science. Although my attitude has always been universalistic, I was particularly drawn to the exact sciences and technology. As a child, I wanted to become an inventor or a physicist. Physics filled me with a sense that reality is marvelous and much more encompassing than meets the eye. I remember the feeling that frequently occurred after having learnt something new, such as Einstein's theories of relativity: a sense of wonder and an expansion of my consciousness that sometimes stayed with me for hours.

The path of mathematical logic. My love for technology resulted in my decision to pursue a master's degree in electrotechnical engineering. While doing so, I was introduced to the beauty of university-level mathematics. I decided to change course towards a master's degree in mathematics. Having embarked on the latter, I asked myself one fundamental question: "what is the essence of mathematics?", and spent many days in the library to seek an answer. This is where I discovered books about the philosophy of mathematics and mathematical logic, and names such as Gödel, Brouwer, Church and Hilbert. Figuratively speaking, I fell in love with these theories and in the end I graduated in a field related to mathematical logic.

The path of technology. Modern technology is another miracle – in which the arrow is turned around: where science is a special kind of projection of the physical world into the conceptual world, technology is a projection from this conceptual world back into the physical world. By making use of our insights in the laws of nature, we can rearrange elements in nature such that it helps us create the things we desire. In a sense, the engineer is a modern wizard who creates 'brews' based on a deep knowledge of these magical laws of nature. I wanted to be part of that process. I was intrigued by computers – not only by their utility, but more so by their inner workings and nature. As a 12-year-old, I took up a job as a newspaper delivery boy and saved money to buy my first computer. In a time where only a few people in each street had such a device. I taught myself programming with relative ease, also in the highly technical low-level machine language. I discovered that I had a strong intuition for creating new technology.

The path of spirituality. In essence, my life revolves around spirituality and (non-dogmatic) religion. It is confusing that there are many interpretations of the words 'religion' and 'spirituality'. I am referring to religion in a 'pure' and abstract sense: one in which it cannot be captured by scriptures, rules or traditions. In this sense, it should also not be confused with *organisations* of religion, such as Christianity. Essential to the abstract interpretation is that it acknowledges there is a pervading awareness and consciousness that is fundamental to everything. This awareness is irreducible to anything else, but, on the contrary, is prior to anything else, and the unifying factor behind all that exists. It lies beyond distinctions and unity, so it is neither one, nor many, but encompasses both. It does not postulate the existence of a God, and neither denies it. We – in our human form – are fragmentary and temporary reflections of this awareness. Such abstract accounts of religion are, among other things, expressed in more mystical teachings such as Advaita Vedanta and Zen Buddhism. One explanation of the etymological origin of the word 'religion' is that it derives from the Latin *ligare* – to connect. In this reading, 'religion' means to 'reconnect'. This closely resonates with the sense in which I use the word – reconnect oneself to one's true, fundamental nature.

The path of science and religion (in the aforementioned sense) are not mutually

exclusive. They complement each other in the pursuit of the truth. A simplified way to put it is that in science awareness is directed to the structure of the outer world as it reaches us through our perceptions, whereas religion is awareness directed to lifting the apparent separation between the observer and the observed, or in other words, consciousness and the object of consciousness. In this interpretation, religion and science should never contradict each other. An analogy may serve as an explanation. One could regard science and religion as two “senses” for pursuing the truth, just as vision and hearing are two different senses for perceiving objects. The sound and appearance of an object are two non-contradicting aspects of that object. On the other hand, they cannot be reduced to each other – hearing and vision are two qualitatively different perceptions. The mainstream divorce of science and religion is a fairly recent phenomenon, probably fueled by the success of science in revolutionising technology. I consider it as a transitory, and probably necessary stage in collective Western culture. Of course, they were never truly divorced. Kepler, Newton, Maxwell and Boole are just a few of many examples of brilliant scientific minds who were deeply religious, and the same holds for many contemporary scientists. Reductionistic science cannot explain things on a fundamental level. This limitation is intrinsic to the scientific method itself, not to the world it observes. I always say: science reduces things we do not understand to simpler things we do not understand. This is a valuable means to increase insight, but can only bring one so far. To go further and penetrate into the essential nature of things, one has to enter the domain of religion – in the sense of ‘re-connection’.

In recent years, there has been a surge of interest in research to explain consciousness. In the view just sketched, this pursuit is destined to fail. Consciousness and awareness are intrinsically out of reach of the reductionistic scientific method. In terms of the above analogy, one cannot see the sound of an object. To perceive the sound of an object, one needs another sense: hearing. Another way to put it is that the scientific method exists within consciousness, and cannot be subjected to it, without this leading to an empty infinite regression. Nevertheless, I believe that the renewed interest is highly valuable, because in the process of pursuing the impossible, the mainstream scientific community is sooner or later explicitly confronted with the shortcomings of its fundamental methods. I expect that in the future, the mainstream scientific method itself will be forced to evolve, and embrace instruments that are nowadays only associated with the more mystical traditions. I emphasise the word ‘mainstream’ here – throughout history there always has been a minority of scientists who already have embraced these means. An example that crossed my path is Henk Barendregt, a well-known logician who is also a teacher in Vipassana meditation. He launched a research program on the border of meditation and neuroscience, already about 20 years ago. To my delight, in recent years, this evolution seems to accelerate.

The path towards wholeness. I believe that we live in a beautiful world. A world

in which collective effort yielded numerous awe-inspiring creations. It has been my desire to contribute to the growth of that collective creativity. However, I also believe we live in a world that is highly *fragmented*. This fragmentation hampers the expression of creativity, and, at worst, it is highly destructive. For example, our exclusive focus on a *fragment* of the ecosystem – economy – destroys species at an unprecedented rate, and endangers humanity's own existence. Most organisations focus on a *fragment* of human traits and capabilities, denying those with other gifts to find their natural place.

The path of building communities with intrinsically valued members. As long as I can remember, I feel a strong bond with the people around me. In particular, I see the beauty of their gifts, and their intrinsic value beyond these gifts. Over the years I witnessed many people, often endowed with remarkable gifts, who do not find their way into society because they do not fit into its predefined pigeonholes. Moreover, beyond one's family, people are predominantly reduced to a means of production. This is exemplified by our labour market, in which job openings are, with a rare exception, formulated in terms of the utility of the candidates for the organisation, and not in terms of their intrinsic value as humans. I think this is a symptom of an underlying deeply alienating view of what it means to be human. It is part of my calling to contribute to people finding their natural place in society, one that is aligned with their natural gifts, their heart, and expresses their intrinsic value as a person beyond their utility.

Just as rivers flow together to form a lake, the mentioned paths in the end all came together in the foundation of a lineage of organisations with the societal mission to foster collective intelligence in service of collective wisdom. This book is, in fact, a spin-off of this lineage and focuses on a predominantly human-technological aspect of the transdisciplinary mission. The current incarnation of the institute is the Nanquanu Institute,

<https://nanquanu.org/>,

which focuses on fostering a world-wide crowd to work on the development of modern collective intelligence systems for a more ecologically balanced, fair and creative society. A collective intelligence system is a system that enhances the collective intelligence of a group of people in a certain respect. The Orcoba-Approach and its concrete extensions developed in this work form examples.

The mentioned paths converge, among other ways, in the following ways in the Nanquanu Institute. (The words related to the paths are *emphasised*.) The systems developed are a sophisticated orchestration of *human* elements and modern *technological* elements. Self-evidently, in collective intelligence, this orchestration requires *building communities*. Nanquanu recruits and values people based on their *intrinsic value*. It has a unique approach to achieve this. Among other things, recruitment

is not based on job descriptions, but on the degree to which people feel resonance with the culture and mission of Nanquanu. If one pursues the realisation of a set of collective intelligence systems that can support humans in as many conditions and disciplines as possible, then it is most effective to develop a set that is as *universally* applicable as possible. Moreover, an organisation could employ such a set to become a '*communitas universalis*' – a *universal community*. The days of the *homo universalis* are long gone due to the knowledge explosion, which makes it impossible for any individual to gain thorough knowledge of all major disciplines. However, a fascinating question posed by Nanquanu is what the smallest community is that would be able to do this. Such communities could contribute to an increased integration, or *wholeness* of knowledge acquisition. *Mathematical logic* is central to one category of these *universal* collective intelligence systems, as has been described in chapter III.3 (p. 151). It is obvious that *learning* lies at the heart of collective intelligence. The *religious* aspect is present in the fact that the collective intelligence systems are created in service of collective wisdom. This, I believe, can only be achieved when continuously reflecting on the place of each system in the ultimate purpose and meaning of life – which is inextricably intertwined with questions of a *religious* nature – in the abstract sense described before. In fact, I consider this reflection itself as an expression of a highly developed collective intelligence.

V.2.3 Acknowledgements

The paths I followed, partially coincided with those of others in ways that, directly or indirectly, left their mark on this work. In particular, I want to express my deep gratitude to the following persons. Louk Fleischhacker was a professor of mathematical logic and the philosophy of mathematical thought at Universiteit Twente, and a person with a beautiful mind and soul. He taught me the intricacies of, among other things, Hegelian and Greek philosophy and his unique and sharp insights in the power and limitations of science that profoundly resonated with mine – and enriched them. I remember my visits to his home, where he, among other things, organised philosophical debates in a small circle, such as on Plato's Parmenides. Unfortunately, he passed away only a few years after I graduated. Nevertheless a part of him will always be in what I have expressed and express. Ronald Siebes is a researcher and developer in the field of artificial intelligence, and a conscientious, eclectic and highly unique individual, with whom I collaborated over an extended period at the Vrije Universiteit Amsterdam. His eclectic mix of interests and activities included, next to scientific research: veganism, drumming, glider aircraft piloting, and stock trading. We had a strong bond – and I deeply valued our conversations that touched almost all aspects of professional and personal

life, ranging from probabilistic automated reasoning, to religion, society and our personal relations. John-Jules Meyer is a professor of multi-agent systems, with whom I feel a strong kinship. So strong, that it sometimes felt as if we could read each other's minds. We collaborated over an extensive period, both at the Universiteit Utrecht (Utrecht University) and at an independent start-up research institute (the Alan Turing Institute Almere). We went through many research and organisational challenges that surrounded this imaginative and risky start-up. This period was an invaluable pressure cooker for me in which I learnt so much about how relations between colleagues in an R&D environment can evolve under high pressure, for better or for worse. For better in the case of John-Jules – who I learnt then, would never betray his colleagues, even under the highest of pressures. John-Jules combines the love for philosophy and abstract symbolic logic, with the love for engineering and the love for people. I have a great admiration for his genuine interest in the people around him, and the way he is of service to their work with his extensive knowledge, bright mind, network and his warm personality. In the past years he was confronted with great challenges in his personal life. What kept him going was that he wanted to be there for his students and colleagues. Koen van der Kruk is a gifted engineer, who I recruited for one of the collective intelligence organisations I founded when he was at the very young age of 15. Already at this age he was a uniquely disciplined individual and a quick and autonomous learner. He contributed considerably to the software implementation of the collective intelligence systems defined in this book. Alle van Meeteren is a man equipped with an eclectic, unique and creative mind. He combines a degree in philosophy and law, but is also an inventor of patented novel computer-interfaces. I consider him as a living example of the fact that flexibility of mind is not age-related, witnessing the speed with which he embraces new technologies and approaches and integrates them in his extensive repertoire of experience and knowledge. In exchange for guidance in advanced technological reflection of his work, he proofread my entire thesis and made valuable contributions to the aforementioned institute: the Nanquanu Institute. Jesse Nortier is a visual artist, with a remarkable insight in aspects of collective intelligence, who is also associated with the Nanquanu network. He did swift and highly professional work in the lay-out design of the cover, and did so wonderfully.

Bregje Theodora Gemma Kleijnen, whose name I always feel inclined to write out fully because of its beauty, and which reflects the spirit of the woman who carries it. I am so grateful for her love and support during the demanding and time-consuming last stage of writing my thesis.

There were also many valuable encounters of a briefer nature. A selection of these were with the following persons. Floris Cohen, brother of the well-known Dutch politician Job Cohen, is a professor of the history of science, whose spirited and philosophically infused lectures in the history of science deepened my insight in the

limits and nature of science. Bas Testerink is a young researcher in the field of multi-agent systems, with a sparkling and imaginative mind, with whom I had numerous fascinating discussions about the nature of consciousness. Dirk van Dalen, Henk Barendregt, Erik Barendsen, Rinus Plasmeijer, Jan-Willem Klop, Marko van Eekelen, Peter Achten, Pieter Koopman, Harold Schellinx, Jaap van Oosten and Jan Kuper are logicians and programming language researchers who contributed to my development in mathematical logic and formal-languages. Jan Visser is another eclectic personality; a physicist, educational scientist and director of the Learning Development Institute, who introduced me to the concept of transdisciplinary learning environments. Thanks to his connections I was able to be a visiting researcher of the renown MIT Media Lab. Willem Kruijer is a statistician at the Wageningen University, who provided valuable pointers for the statistical aspects of my thesis. Geoff Sutcliffe is a professor of automated theorem proving at the University of Miami, and one of the binding forces behind the associated research community. He provided me with valuable tips about what reasoners to use for my purposes, and participated in experiments. Stephan Schulz is a professor at the University of Duisberg-Essen, and the developer of automated theorem prover Eprover that I integrated into SWiFT, and he also participated in my experiments. Koen Claessen is a professor at Chalmers University of Technology, whose reasoner Paradox I integrated into SWiFT, and with whom I had delightful conversations when he was visiting Utrecht University a few years ago. He had a good grasp on several caveats related to acquiring formal-fluency, one of the main topics of this book. These could serve as future challenges to be tackled with SWiFT. Peter Patel-Schneider is a leading expert in description logic, with whom I had a valuable exchange about the theoretical limits of the application of automated theorem proving in daily life. Gerko Vink is a statistical scientist at Utrecht University, who generously helped me with applying advanced features of the statistical package R. Dirk Thierens and Krzysztof L. Sadowski are scientists in the field of evolutionary computation, who provided valuable tips about what evolutionary computation techniques to apply to the systems in this book. Dirk, who maintains an extensive network in his field, additionally provided accurate recommendations for scientists who could be particularly interested in the human-technological hybridisation of evolutionary computation proposed in this book. Catarina Dutilh Novaes is a philosopher who organised a reading group, which deepened my insight in the philosophy of language, amongst which Wittgenstein's works. Johan van Benthem is a professor in dynamic logic, who participated in the interview cycle I initiated during my master's study about the psychology of practicing mathematics, which, among other things, deepened my insight in the diversity of every day practice of scientific research and the way to access your mathematical creativity on a psychological level. Pieter Koopman is a programming language researcher, who provided valuable functional programming tips during the

implementation of the parsers needed for SWiFT.

I am also indebted to the many persons who participated in the experiments, or made other small contributions. To name a few: Jesse Nortier, Zhisheng Huang, Szymon Klarman, Laura Hollink, Miriam de Koning, Jan Linnebank, Alexandra Huisman, Jelle Brands, Alexia Hageman, Sandjai Bhulai, Michel Klein, Marloes Dirkx, Roeleke Schepers and many more.

My deepest gratitude goes to the following persons. Marre Groenouwe is a mystically inclined artist draftswoman. She lives a modest life with an unwavering dedication to fostering a profound connection with the essential truths of life. She is a heart-warming and unsung hero. Ebere Groenouwe is a mixed media artist, with profound philosophical insights in life that permeate her work and her way of life. She is among the most courageous people I know. Neither of them are active in the field of science or technology. Yet, I feel that a great part of my work consists of translating their insights in life into the domain of my particular gifts.

Glossary

If a term is defined in the book, the pages on which it is defined are displayed distinctively. An example: ‘110’ is an ordinary page reference, while ‘117’ refers to the page with the definition.

acceptance set 94, 96, 105

acceptance-by-percentage-deviation-from-mean 96, 105

ACE 146

algorithmic-construction-stage 217, 256, 257, 292, 296

algorithmic-defence-round 217, 229, 237, 238, 247, 248, 255, 256, 291, 295, 307–310, 324, 326

algorithmic-execution-stage 217, 256, 258

ancillary-metric-for-integration-of-automated-deduction 165, 166–168

answer-language 263, 264, 265, 289, 293, 300

approximate-metric 169, 218

artificial grammar 144, 148

artificial grammar learning 144, 147, 148, 149

artificial-creatability 60

assumption-set 163, 174, 180, 188, 189

ATP Abbreviation for automated theorem proving. 153

attractor The meaning of the word ‘attractor’ in the context of cybernetics. 41

- automated theorem prover** [138](#), 365
- automated theorem proving** 13, 153, 310, 349
- automated deduction** iv, 4, 13, 34, 136, [137](#), 138–140, 146, 167, 220
- automated deductive reasoner** 13, 131, 134, 135, [138](#), 140, 141, 143, 151, 155, 165, 169–175, 180, 183–186, 189–192, 194, 198, 199, 209, 211, 212, 221, 226, 235, 243, 246, 264, 300, 303, 308, 309, 316, 361
- automatically deducible fraction** [179](#), 180, 190, 218, 228, 237, 298, 305, 309, 310
- better expressivity for human usage** [182](#)
- branch** In the given context an abbreviation for branch [112](#), 113–115, 117, 120, 121, 124, 350
- branch-continuation** [112](#), 114
- branch-creator** [112](#), 113–115
- branch-root** [112](#), 114
- Brasic language** [286](#), 287, 290, 291, 294, 295
- bridge construction round** 248, [250](#), 254
- bridge language** [263](#), 286, 288, 290, 294
- c factor** [29](#), 30, 35
- capability-composition** [8](#), 43–46
- capability-constitution** 35, [44](#), 45, 46, 147, 352
- capability-specific Orcoba extension** [55](#), 304
- CBS** Abbreviation for Constitution-Based-Subleme 201, [228](#), 229
- Challenge-Based-Coba** [55](#), 57, 63
- Challenge-Based-Orcoba** 47, [55](#), 58, 109, 126, 194, 211, 305
- challenge-assessment** [61](#), 226
- challenge-set** [57](#), 58, 60, 61, 194, 213, 220, 263, 289, 293, 299, 305

- child-box** [53](#)
- child-creator** [112](#), 113–115
- Chomsky hierarchy** [148](#)
- class-of-proportional-selection-operators** [73](#)
- class-of-systems** 49, [52](#), 53, 160
- Coba** [43](#), 46, 47, 54, 55, 57, 58, 63
- Coba-Approach** [43](#), 44
- cognitive diversity** [9](#), 10
- collective intelligence** 9, [23](#), 24, 27–29, 31, 35, 36, 38, 40, 41
- collective IQ** [29](#)
- collective-formal-thinking** [198](#), 306
- community-pursuing-a-capability** 6, [8](#), 10, 43, 44, 125, 335
- complete** In the given context, the meaning of ‘complete’ as described in complete-metric 157, 165, 166, 170
- complete-metric** 156, [157](#), 158, 167, 351
- complex system** 30–32
- composite-node** [205](#)
- composition-record-user** [43](#), [50](#), 55, 57, 78, 116, 118
- composition-record** 9, [10](#), 11–13, 15, 21, 33, [43](#), 44–47, 50, 51, 55, 57, 61–66, 78–80, 113, 114, 116–127, 143, 187, 193, 194, 199, 201, 206, 209–213, 218, 227, 228, 235–237, 239, 241, 243, 248–251, 259, 261, 299, 304–307, 316, 317, 323, 324, 326, 327, 330, 331, 335
- compositionality of semantics** [24](#)
- Computer-Supported Collaborative Learning** [36](#)
- Computer-Supported Collaborative Work** 200
- Computer-Supported Cooperative Work** [36](#)

consti-study-round [248](#), 250, 251

constitution Abbreviation for capability-constitution 44, 45, 122, 217

Constitution-Based-Subleme 199, 201, [228](#), 229, 350

constitutional-aspect [44](#), 45, 210

context-free grammar [148](#), 268

contributive-aspect [44](#), 45, 209

contributive-capability-record [45](#)

controlled natural language [146](#)

crossover In the given context, the meaning of ‘crossover’ as described in evolutionary computation 124

cubic spline interpolation [107](#), 119

curve fitting [99](#), 100, 101, 103, 106, 108, 119

cybernetics [30](#), 31, 32, 41

DBpedia <https://wiki.dbpedia.org/> 136

decentralised-development [9](#), 46, 47, 125, 335

deduction-algorithm [217](#), 219–221, 234–238, 247, 248, 255, 256, 306–308

deductive reasoning 131, [137](#)

deductive-closure [177](#), 198

deductive-formal-language 4, 33, 133, [138](#), 144, 146, 149, 197, 354

dependent variable [97](#)

dependent-capability-graph [62](#), 127

digital-entity [202](#), 203

discrete-time series [91](#)

discrete-time stochastic process [102](#)

- domain-related-semantic-conditions** 185, 186, 187, 225
- EC** Abbreviation of evolutionary computation 66
- educational science** 35, 109, 119
- EERG** 190, 202, 203, 205, 206, 208, 212, 213, 230, 231, 310, 317, 318, 321, 322
- Efe-language** 270, 289–291
- elementary-node** 205, 209, 235
- Elo rating system** 109, 123
- Eprover** 262, 280, 281, 347
- equilibrium** The meaning of the word ‘equilibrium’ in the context of cybernetics.
41
- equivalent** 176–178
- equivalent-in** 178
- evo-individual** 76, 77, 78, 80, 85, 110, 112, 114, 118, 123
- Evohut** 65, 66, 76, 77, 78, 80–82, 85, 86, 118
- evolutionary computation** 21, 31, 65, 66, 67–73, 76–78, 110, 113, 117, 124, 352, 353, 355, 360, 361
- evolutionary computation-algorithm** 69, 70, 73
- evolutionary human-technological system** 67
- Exact-Entity-Relation-Graph** 202, 229–231
- explanation-degree** 46
- Externally-Open-Coba** 46, 47
- fitness evaluation** 67, 70, 78, 80, 84, 118, 120, 121, 123
- fitness function** 31, 70, 71, 72, 78, 80, 82, 114, 127
- fitness proportionate selection** 70, 71, 123
- fitness score** 70, 78, 122, 362

fitness-evaluation-space 72

flatline-estimation 103, 104–106, 119

flatline-level 82, 84–87, 89, 90, 103, 104, 106–108, 119

flatline-moment 82, 84–87, 89, 103, 104, 106

fluency-player 217, 241, 245

focus The meaning of the word ‘focus’ in the context of fostering states of affairs.
163

FOL Abbreviation for first-order logic 184, 188, 264, 266–268, 270, 271, 275–277, 286, 287

Folminquon-language 274, 275, 276, 293–295

Folnuminquon-language 276, 277, 278, 281, 293, 296

Folsequa-language 270, 271, 272, 289

formal grammar 144, 147, 148

formal language In the given context, the meaning of ‘formal language’ as described in formal language syntax theory 148

formal-language-complexity 183

formal language syntax theory 148, 354

formal language theory 138, 147, 148

formal-challenge 60

formal-fluency Abbreviation for human-fluency-in-deductive-formal-languages iv, v, 3, 4, 7, 8, 11, 13–17, 33, 41, 42, 56, 109, 122, 131, 132, 133, 134–137, 140, 141, 143–147, 151, 155, 187, 193, 194, 208, 210, 212, 215–217, 220–223, 226, 238, 240, 242, 245–247, 259, 262, 297–299, 301, 304–309, 313, 326, 336, 339, 340, 347

formal-language In the context of this book abbreviation for deductive-formal-language, unless another interpretation is specified at the location where the term occurs. 6, 12, 14–16, 34, 56, 86, 134–136, 138, 140–142, 144, 146–149, 166, 171, 172, 174–189, 191–194, 197, 198, 201, 202, 209, 211, 212, 214, 215, 217, 221, 223, 225, 227, 229, 231, 232, 236, 242, 247, 248, 250, 255, 256, 262–266, 286–288, 292, 296, 299, 300, 305, 308–310, 340, 347

formal-language-for-automated-deduction iv, 4, 6, 138, 147

formal-super-language 171, 172, 173

formal-target-language 221, 223–226, 228, 229, 242, 245, 246, 248, 250, 263, 289, 293, 297, 305, 306, 308, 310

fully-automated-deduction 138

g factor 27, 29

game theory 31, 32

game with a purpose 215

Game-Based-Orcoba 43, 55, 59, 63, 78, 126, 217, 241

game-coordinator 235, 236, 308

general intelligence 8, 27

general-Orcoba-extension 55, 304

general-local-deductive-closure 177

generalised-quantifier 276

generation cycle 69, 73, 117, 118, 123

generic-properties 53, 54

genotype In the given context, the meaning of ‘genotype’ as described in evolutionary computation 68, 69, 70, 71

Global Brain Institute <https://sites.google.com/site/gbialternative1/>
30

global selective fitness function 72

growth-estimation 103, 106–108, 119, 123

higher-order logic 141, 142

higher-order-Coba 58, 59

higher-order-Orcoba 59, 61, 62, 64, 116, 117, 126, 127, 223, 246, 298, 306, 339

- higher-order-SWiFT** 131, 195, 223, 224, 246, 247, 248, 261, 298, 299, 305, 306
- hucolligence** Abbreviation for human-collective-intelligence 6, 7, 8, 9, 12, 14, 16, 21, 23, 26, 31, 35, 39–42, 47, 48, 50, 66, 76, 123, 125, 126, 335, 336, 339
- human swarming** 38
- human-based genetic algorithm** 76
- human-artefact-co-evolution** 9, 12, 44, 47, 125, 335
- human-capability-enhancing-Evohut** 76
- human-collective-intelligence** iii–v, 3, 4, 6, 7, 14, 21, 23, 24, 25, 38, 42, 125, 335, 356
- human-crossover** 124, 127
- human-fluency-in-formal-languages-for-automated-deduction** iii, iv, 3, 7, 131, 133, 143, 336, 356
- human-fluency-in-deductive-formal-languages** Abbreviation for human-fluency-in-formal-languages-for-automated-deduction. 141, 354
- human-relational-antonym** 288
- human-technological-fitness-evaluation** 77
- hurelan-construct** 288, 293, 294
- Hurelan-language** 288, 294, 295
- ideal-reasoner-oracle** 171, 173
- ideal-translator-oracle** 171, 172, 173, 188
- independent variable** 97
- institutionalised-self-reflection** 9, 30, 46, 47, 125, 335
- integration-of-automated-deduction** 131, 133, 140, 141, 143, 146, 151, 152–156, 159, 160, 162, 166–169, 171, 172, 181, 184, 189, 192, 197, 215, 236, 303, 310
- integration-of-automated-reasoning** 152, 155
- intention** The meaning of the word ‘intention’ in the context of fostering states of affairs. 156

- interactive evolutionary computation** [76](#)
- Internally-Open-Coba** [46](#), 47
- intersect-under-equivalence** [178](#)
- iteration** [69](#), 110
- join-graph** [106](#), 107
- join-interval** [106](#), 107, 108
- knowledge engineer** 136
- knowledge representation and reasoning** 13, [139](#), 155, 180, 183, 218, 304, 310
- knowledge-engineering** 218, 292
- known-formal-language-complexity** [183](#), 184
- KR-base** 11, 12, 34, 56, [142](#), 169, 170, 172, 183, 184, 188–192, 194, 198, 201–203, 211, 212, 228, 264–266, 273, 281, 310, 340
- KR-language** 142, 185, 187, 230, [264](#), 265, 300, 306, 361
- layered-system-approach** [52](#), 192
- learning curve** [82](#)
- line plot** [100](#)
- Linked Open Data** https://en.wikipedia.org/wiki/Linked_data [136](#)
- local selective fitness function** [72](#), 73
- local selective fitness function generator** [73](#)
- local-automatically-deducible-fraction** [178](#), 179
- local-deductive-closure** [177](#), 178
- locally weighted polynomial regression** [106](#)
- LOESS** [106](#), 119
- longitudinal data analysis** [102](#), 103

MAS Abbreviation for multi-agent system. 34, 38

method of the least squares 99

metric The meaning of the word ‘metric’ in the context of fostering states of affairs.
151, 152, 154, 156, 157–159, 162, 163, 164–170, 173, 174, 179, 183, 189, 191, 192,
218, 238

metric-for-integration-of-automated-deduction- α 168

class-of-metrics-for-integration-of-automated-deduction 167, 168

metric-for-integration-of-automated-deduction 165, 167–169, 173, 174, 184, 185,
188, 189, 192

metric-for-value-of-questions 165, 166, 167, 168, 179

micro-macro link 32

minimally-self-evidently-valuable-state-of-affairs 155, 157, 159

MIT Center for Collective Intelligence <https://cci.mit.edu/> 29, 30

mostInfo-construct 266, 273, 280, 281

multi-agent system 32, 34, 38, 358

normative multi-agent system 34, 35

normative system 34, 35

NoUna 292

NP complete 280

number-quantifier 276, 277, 278, 281

objective fitness function 71, 72, 75, 77, 81, 82, 114

ontology 136, 200, 211

operational In the given context, the meaning of ‘operational’ as described in
operational-metric 157, 158, 166, 169, 170, 174

operational-metric 156, 157, 158, 218, 358

- Or-evohut** [55](#), 59, 65, [76](#), 109, 110, 112, 113, 118, 120, 121, 126, 127, 194, 211, 213, 305–307, 310, 336, 339, 340
- Or-evohut- α** 75, 107, [110](#), 111–115, 117, 120–124, 126, 127, 313, 339
- Orcoba** [10](#), 11–13, 15, 17, 30, 33–35, 41, 43, [47](#), 48–50, 54–56, 58, 59, 61, 63–66, 76, 77, 122, 125–127, 194, 304–306, 318, 335, 336, 339
- Orcoba-Approach** 9, [10](#), 11–13, 15, 16, 21, 30, 31, 33, 35, 43, 47–50, 65, 66, 76, 117, 124–127, 131, 143, 147, 193, 194, 199, 211, 213, 215, 217, 304–307, 313, 335, 336, 339, 344
- Orcoba-extension** [50](#), 103, 199, 201
- orcoba-fosterer** [50](#), 61
- organisational psychology** 30, [35](#)
- organisational studies** 30, 35
- OWL** Abbreviation for Web Ontology Language. [142](#), 186, 187, 220, 227
- OWL2** Abbreviation for the 2nd version of Abbreviation for Web Ontology Language.. [142](#)
- Paradox** [262](#), 280–282, 347
- parent-box** [53](#)
- partially-automated-deduction** [138](#)
- pattern** [265](#)
- pattern variable** [265](#), 266, 272, 273, 278, 281
- pattern-language** 264, [265](#), 266, 272, 278, 300
- peer-to-peer** 33
- performance metric** 109
- performance-function** 76, [80](#), 81–87, 89–91, 101, 103, 104, 106, 118, 119, 122, 123
- performance-function-estimation** [103](#), 107
- performance-function-estimator** [103](#), 109, 123, 127

performance-function-estimator- α 103, 105, 106, 107, 123

performance-function-measurement 91, 103

performance-index 81, 82, 84–87, 118, 119

performance-level 76, 80, 85, 87, 103

performance-measurement 87, 91, 103, 104

performance-sample-data 91, 104, 107

phenotype In the given context, the meaning of ‘phenotype’ as described in evolutionary computation 69, 70, 72

physiological synchrony 36

Plonu-language 278, 279

Plonumo-language 279, 280, 281, 293–296

Plosequa-language 272, 273, 292

Plosequamo-language 273, 289, 290

population In the given context, the meaning of ‘population’ as described in evolutionary computation 69

population The meaning of the word ‘population’ in the context of statistics. 92, 93

population parameter 92, 104

potential-child-creator 110, 114

potential-child-realisation 112

potential-child 110, 111–115

potential-evo-individual 110

power set 94

proof-length 177

proper-extension 52

psychometrics [27](#)

purely·human·fitness·evaluation [77](#)

pursued-capability [8](#), 10, 43, 50, 51, 55, 57, 60, 61, 63, 199, 305

query-language 142, [264](#), 265, 280, 300

question-formal-language 255, [263](#), 265, 289, 290, 293, 294

question-attack-round [217](#), 247, 248, 255, 256, 259, 292, 296, 297, 309, 324, 326

R 106, 107, 347

Racer 220

rank based selection [70](#), 123

RDF <https://www.w3.org/RDF/> [142](#), 186, 226, 227, 232, 234, 361

RDF+S In this book used as an abbreviation for RDF (<https://www.w3.org/RDF/>) and RDFS (<https://www.w3.org/TR/rdf-schema/>), two well-known standard KR-languages that complement each other and intended to be used jointly. They were developed in association with W3C. 136, [142](#), 146, 186, 187, 190, 224, 226–230, 232, 306, 331

RDFS <https://www.w3.org/TR/rdf-schema/> [142](#), 226, 227

real-time·collective·formal·thinking [11](#), 56, 131, 194, [198](#), 199, 208, 210–213, 305, 306, 310, 340

reasoner In the given context, the meaning of ‘reasoner’ as described in automated-deductive-reasoner 165, 169, 171, 184, 187, 188, 191, 226, 227, 246, 280

reasoner-chooser-function [189](#)

reasoner-chooser-oracle [172](#), 189

reasoning-goal 171, 172, 188, 189, [218](#), 281

recombination In the given context, the meaning of ‘recombination’ as described in evolutionary computation [124](#), 127

composition-record-developer [43](#), 46, 50, 63, 117, 124, 126, 205

- composition-record-player** 43, 55, 78, 84, 86, 103, 117, 123, 124, 127, 217
- regression analysis** 97, 98–100, 103, 106, 119
- regression function** 97, 98
- regression splines** 100
- regular grammar** 148
- relational antonym** 288, 294
- relevant-assumption** The meaning of the word ‘relevantAssumption’ in the context of fostering states of affairs. 163, 188
- relevant-property** The meaning of the word ‘property’ in the context of fostering states of affairs. 163
- repeated measures** 102
- representation space** 69
- requirement** The meaning of the word ‘requirement’ in the context of fostering state of affairs. 163, 164, 180
- score** In the given context, abbreviation for fitness score 70
- second-order cybernetics** 31
- selection operator** 70, 75, 123
- selective fitness function** 71, 72, 73, 75
- selective pressure** 70, 71
- self organisation** 28
- self-evidently-valuable-state-of-affairs** 155, 156, 158
- Semantic Network** 198, 202
- Semantic Web** 142, 145, 146, 153, 199, 200, 202, 203, 215, 216, 234, 237, 304
- Semantic Web Stack** 153, 162
- Semantic Wiki** 136, 147, 200, 216

- sentence** The meaning of the word ‘sentence’ in the context of predicate logic. For more details, see: [https://en.wikipedia.org/wiki/Sentence_\(mathematical_logic\)](https://en.wikipedia.org/wiki/Sentence_(mathematical_logic)). 188
- sentence-equivalence** [176](#), 178
- sequential analysis** [93](#), 94, 103, 104, 118, 119
- sequential procedure** [93](#), 95, 96
- Sigma scaling** 65, 74, [75](#)
- smoothing** [100](#), 101
- social choice theory** 38
- social psychology** 35
- social simulation** [31](#), 32–34
- solution space** [69](#), [71](#), 76, 78
- spacing effects** 127
- SPARQL** [142](#), 220, 227, 236–238, 324, 328
- spline** [100](#), 101, 107
- stationary** The meaning of the word ‘stationary’ in the context of statistics. 104
- statistical estimation theory** [92](#), 93, 104
- Stochastic Universal Sampling** [74](#)
- structural-semantic-conditions** [185](#), 186, 187, 223, 225, 242
- structural-sentence-compatibility** [175](#)
- structurally-compatible** [175](#), 176
- structure** The meaning of the word ‘structure’ in the context of first-order logic. [267](#)
- Student’s t-distribution** [95](#)
- sub-Orcoba** [62](#), 117, 127

sufficiently-expressive-for-human-usage 180, 181–183

swarm intelligence 37, 38

SWiFT v, ix, 10–12, 17, 56, 77–80, 103, 122, 131, 194, 195, 208, 209, 211, 215, 216, 217, 219, 221–223, 225, 228, 240, 242, 243, 246–248, 250, 255, 297–299, 305, 307–310, 313, 316, 340, 347, 348

SWiFT-Efe 248–259, 263, 288, 289, 290, 292, 293

SWiFT-NoUna 256, 262, 263, 281, 292, 293, 294, 296

SWiFT-Base 217, 218, 220, 222–225, 235

SWiFT-Fixtal 33, 131, 195, 222–224, 225, 226, 241, 242, 245, 246, 297, 305–307, 339

SWiFT-Fixtal- α 122, 223, 224, 225, 226, 227, 235, 238–240, 242, 243, 307, 308, 323

SWiFT-Focused 122, 131, 195, 222–224, 246, 247–249, 255, 256, 259, 262–264, 281, 288, 297–301, 305, 307

SWiFT-Focused repository 282, 285

SWiFT-Full 131, 195, 216, 219, 220, 221–226, 241, 242, 245–248, 256, 263, 297, 298, 305–308, 310, 339

SWiFT-class-of-games 56, 59, 194, 215, 216, 219, 223, 226, 246, 247, 305–307, 309, 310, 339

SWiFT-Turtle 228, 229–232, 235, 236, 242, 323, 324

syllogistic logic 137

symbol grounding problem 24

symbolic-language 175, 177–179

syntactically formal language 148

System- α -for-real-time-collective-formal-thinking 122, 194, 198, 229, 305, 315, 365

system-for-realcolforthi Abbreviation for system-for-real-time-collective-formal-thinking. 56, 58

system-for-real-time-collective-formal-thinking 56, 59, 364

System-Real- α Abbreviation for System- α -for-real-time-collective-formal-thinking
11, 17, 58, 198, 199–201, 208, 211, 212, 232, 233, 310, 315

takeover time 70, 71

technologically enhanced learning 109

term-set 274, 276

text-assignment 236, 308, 309

text-generation-language 263, 289, 293

the Nanquanu Institute 344, 346

theorem prover Commonly used abbreviation for automated theorem prover. 138

time series 91, 98, 99, 101, 102, 104

time series analysis 91, 98, 99, 101, 102, 103

time-consuming-reasoner-oracle 171, 172, 180

tournament based selection 70, 123

TPTP language 142, 281

TPTP world 152, 153

translation-duration 259

translation-round 217, 248, 250, 252, 253, 309

translation-session 309

Turtle 228, 229, 231

Uniform Resource Identifier 142, 231, 365

unique name assumption 234, 281, 290, 292, 293, 295

unrestricted grammar 148

URI Abbreviation for Uniform Resource Identifier. 142, 203, 230, 231, 235

vague-metric 157

value [161](#)

W3C Abbreviation for World Wide Web Consortium. [145](#), 361

Web Ontology Language See <https://www.w3.org/OWL/>. [142](#), 359

weighted·automatically·deducible·fraction [179](#), 180, 218

weighted·local·automatically·deducible·fraction [179](#)

well-defined In the given context, the meaning of ‘well-defined’ as described in
well-defined·metric 157, 158, 165, 169, 174

well-defined·metric 156, [157](#), 366

well-formed formula 274

wisdom of the crowd [36](#), 37, 38

World Wide Web Consortium <https://www.w3.org/>) [145](#), 366

Curriculum vitae

For almost two decades, Chide Groenouwe's main occupation has been fostering human collective intelligence in service of collective creativity and collective wisdom. Within this transdisciplinary challenge, he focuses on the co-evolution of human and technological elements. He does so by means of R&D into *collective intelligence systems*: systems that consist of a clever orchestration of human and technological elements with the purpose of enhancing the collective intelligence of a human group in a certain respect. Associated keywords are technically augmented human collective intelligence, human swarming, socio-technological systems, wisdom of the crowd, intelligence amplification, collective IQ, human computation, distributed cognition, social software and crowdsourcing. Among other things, such systems may enhance aspects within the practice of science, art, engineering, journalism, and they may enhance aspects within the social, political, economical and ecological dimensions of society.

In 2001, Chide graduated from the faculty of applied mathematics at the Universiteit Twente, Netherlands (cum laude). His Master's programme consisted of a self-composed program ("vrij doctoraal") in mathematical logic, in collaboration with Universiteit Utrecht (Utrecht University) and the Radboud Universiteit, both having a strong international reputation in logic. He studied, among others with Dr Louk Fleischhacker, Dr Jan Kuper, Prof Dirk van Dalen, Prof Henk Barendregt, Prof Erik Barendsen and Prof Rinus Plasmeijer. From 2001 to 2003 he worked as an educator in computer science and mathematics at the Hogeschool van Utrecht, one of the largest colleges in the Netherlands. In parallel, he founded Network Universalis, an organisation devoted to R&D into collective intelligence. After a short period as junior researcher in formal distributed knowledge representation and reasoning with the department of computer science of the Vrije Universiteit Amsterdam (Prof Frank van Harmelen, Prof Guus Schreiber and Dr Ronald Siebes), he continued in 2005 at the same institute with research into collective intelligence systems to enhance the collaborative practice of science (Prof Jan Top). From 2010 to 2014 he worked at the Alan Turing Institute Almere (Prof J-J. Ch. Meyer), an institute devoted to

the development of hybrid human-artificial intelligence systems to support medical practitioners, doing research in how to foster technically augmented human collective intelligence among collaborating scientists, engineers and medical practitioners. Here, he was the originator of the Marama research project, a visual programming language for bridging the gap between programmers and non-programmers in collaborative software system construction. From 2011 to 2014, he was also leader of Ba-ila, the successor of Network Universalis. In pursuit of additional funding for Marama, he joined an initiative led by Prof Rinus Plasmeijer (Radboud Universiteit) to co-write a larger interinstitutional research grant application related to technologically augmented collective intelligence. The application succeeded. This resulted in 2014 in his part-time research position at Utrecht University, with the group Intelligent Systems of Prof J-J. Ch. Meyer. This enabled him to continue working on Marama and get involved in Task-Oriented Programming (Rinus Plasmeijer et al), a novel software development paradigm to enhance the collective intelligence of people working towards a common goal. In parallel, he recently finished writing a PhD thesis in the field of technically augmented collective intelligence. From 2014 up to the present, he is director of the Nanquanu Institute for collective intelligence.

Bibliography

- Abele, A., McCrae, J. P., Buitelaar, P., Jentsch, A., and Cyganiak, R. (2019). Linking open data cloud diagram 2017. <http://lod-cloud.net>, retrieved January 6th, 2019.
- Ahlberg, J., Nilson, E., and Walsh, J. (1967). The theory of splines and their applications. *Mathematics in Science and Engineering*, New York: Academic Press, 1967.
- Ahn, von, L. (2006). Games with a purpose. *Computer*, 39(6):92–94. <http://csdl.computer.org/dl/mags/co/2006/06/r6092.pdf>.
- Alexander, J. C. (1987). *The micro-macro link*. Univ of California Press.
- Anastasi, A. (1992). What counselors should know about the use and interpretation of psychological tests. *Journal of Counseling & Development*, 70(5):610–615.
- Aristotle (350BCEa). *Metaphysics*. The Internet Classics Archive. <http://classics.mit.edu/Aristotle/metaphysics.html>.
- Aristotle (350BCEb). *The politics of Aristotle*, volume 1. Clarendon Press. <http://classics.mit.edu/Aristotle/politics.html>.
- Arkes, H. R. and Blumer, C. (1985). The psychology of sunk cost. *Organizational Behavior and Human Decision Processes*, 35(1):124 – 140. <http://www.sciencedirect.com/science/article/pii/0749597885900494>.
- Arlinghaus, S. (1994). *Practical handbook of curve fitting*. CRC press.
- Arora, S. and Barak, B. (2009). *Computational complexity: a modern approach*. Cambridge University Press.
- Ashby, W. R. (1947). Principles of the self-organizing dynamic system. *The Journal of general psychology*, 37(2):125–128.

- Ashby, W. R. (1956). *An introduction to cybernetics*. Chapman & Hall, London.
- Axelrod, R. (1997a). Advancing the art of simulation in the social sciences. In *Simulating social phenomena*, pages 21–40. Springer.
- Axelrod, R. (1997b). *The complexity of cooperation: Agent-based models of competition and collaboration*, volume 3. Princeton University Press.
- Baader, F. (2003). *The Description Logic Handbook*. https://www.researchgate.net/profile/Deborah_Mcguinness/publication/230745455_The_Description_Logic_Handbook_Theory_Implementation_and_Applications/links/0deec51cfb6d8ae9d3000000.pdf.
- Bäck, T. (1994). Selective pressure in evolutionary algorithms: A characterization of selection mechanisms. In *In Proceedings of the First IEEE Conference on Evolutionary Computation*, pages 57–62. IEEE Press.
- Bäck, T., Fogel, D. B., and Michalewicz, Z., editors (1997). *Handbook of Evolutionary Computation*. IOP Publishing Ltd., Bristol, UK, UK, 1st edition. <http://portal.acm.org/citation.cfm?id=548530>.
- Baker, J. E. (1987). Reducing bias and inefficiency in the selection algorithm. In *Proceedings of the 2nd International Conference on Genetic Algorithms, Cambridge, MA, USA, July 1987*, pages 14–21.
- Balke, T., da Costa Pereira, C., Dignum, F., Lorini, E., Rotolo, A., Vasconcelos, W., and Villata, S. (2013). Norms in mas: Definitions and related concepts. In *Dagstuhl Follow-Ups*, volume 4. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.
- Barendregt, H. P. (2014). private communication.
- Bauer, T. N., Truxillo, D. M., and Erdogan, B. (2015). *Psychology and work: Perspectives on industrial and organizational psychology*. Routledge.
- Becket David, Berners-Lee, T. (Version of March 28th, 2011). Turtle - terse rdf triple language. <https://www.w3.org/TeamSubmission/turtle/>.
- Bemer, R. W. (1971). A view of the history of cobol. *Honeywell Computer Journal*, 5(3):130–135.
- Beni, G. and Wang, J. (1989). Swarm intelligence in cellular robotic systems, proceed. nato advanced workshop on robots and biological systems, tuscan, italy, june 26-30. *Applied Mathematics and Computation*, 3:268–308.

- Berners-Lee, T. (1999). *Weaving The Web*. HarperCollinsPublishers.
- Berners-Lee, T. (2001). The semantic web. *The Scientific American*.
- Bibel, W. (2013). *Automated theorem proving*. Springer Science & Business Media.
- Bjork, R. (2017). Creating desirable difficulties to enhance learning. *Progress*.
- Bledsoe Jr, D. (2011). Handwriting speed in an adult population. *Advance for Occupational Therapy Practitioners*, 27(22):10.
- Bobzien, S. (2016). Ancient logic. In Zalta, E. N., editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, winter 2016 edition. <https://plato.stanford.edu/archives/win2016/entries/logic-ancient>.
- Boella, G., Van Der Torre, L., and Verhagen, H. (2006). Introduction to normative multiagent systems. *Computational & Mathematical Organization Theory*, 12(2-3):71–79.
- Boole, G. (1847). *The Mathematical Analysis of Logic*.
- Boole, G. (1854). *An Investigation of the Laws of Thought*. Cosimo classics. Lightning Source Incorporated.
- Boring, E. G. (1923). Intelligence as the tests test it. *New Republic*, pages 35–37.
- Bowen, P., Debreceeny, R., Rohde, F., and Basford, J. (2006). Selecting optimal instantiations of data models—theory and validation of an ex ante approach. *Decision Support Systems*, 42(2):1170 – 1186. <http://www.sciencedirect.com/science/article/B6V8S-4HYN4XH-1/2/9f248825e74dec153cac3454dfd0dad2>.
- Bry, F., Schaffert, S., Vrandečić, D., and Weiland, K. (2012). Semantic wikis: Approaches, applications, and perspectives. In *Reasoning Web International Summer School*, pages 329–369. Springer.
- Buckingham, S. (2007). Modelling naturalistic argumentation in research literatures: Representation and interaction design issues. *International Journal of Intelligent Systems*, (22).
- Cambridge Dictionary (2019). fluency. <https://dictionary.cambridge.org/dictionary/english/fluency>, retrieved January 6th, 2019.
- Cañas, A. J., Hill, G., Carff, R., Suri, N., Lott, J., Gómez, G., Eskridge, T. C., Arroyo, M., and Carvajal, R. (2004). Cmaptools: A knowledge modeling and sharing environment. <http://eprint.ihmc.us/89/1/cmc2004-283.pdf>.

- Carnap, R. (1956). *Meaning and Necessity*. The University of Chicago Press.
- Carnielli, W. A. and Marcos, J. (2001). Ex contradictione non sequitur quodlibet. *Bulletin of Advanced Reasoning and Knowledge*, 1:89–109. <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.62.2975&rep=rep1&type=pdf>.
- Chambers, F. (1997). What do we mean by fluency? *System*, 25(4):535–544.
- Chella, A. and Manzotti, R. (2007). Artificial intelligence and consciousness. In *Association for the advancement of Artificial Intelligence Fall Symposium*, pages 1–8.
- Chomsky, N. (1959). On certain formal properties of grammars. *Information and Control*, 2(2):137 – 167.
- Cipolla, C. (1969). *Literacy and Development in the West*.
- Claessen, K. and Sörensson, N. (2003). New techniques that improve mace-style finite model finding. In *Proceedings of the CADE-19 Workshop: Model Computation-Principles, Algorithms, Applications*, pages 11–27. Citeseer.
- Cleveland, W. S. (1979). Robust locally weighted regression and smoothing scatterplots. *Journal of the American statistical association*, 74(368):829–836.
- Cleveland, W. S. and Devlin, S. J. (1988). Locally weighted regression: an approach to regression analysis by local fitting. *Journal of the American statistical association*, 83(403):596–610.
- Collste, G. (1992). Expert systems in medicine and moral responsibility. *Journal of Systems and Software*, 17(1):15 – 22. <http://www.sciencedirect.com/science/article/pii/016412129290076V>.
- Colyvas, J. A. (2012). Performance metrics as formal structures and through the lens of social mechanisms: When do they work and how do they influence? *American Journal of Education*, 118(2):167–197. <https://doi.org/10.1086/663270>.
- Cook, S. A. (1971). The complexity of theorem-proving procedures. In *Proceedings of the Third Annual ACM Symposium on Theory of Computing*, STOC '71, pages 151–158, New York, NY, USA. ACM. <http://doi.acm.org/10.1145/800157.805047>.
- Coolen, M. (1992). *De Machine Voorbij*. To our knowledge only available in Dutch. Translation of title: Beyond the Machine.

- Copeland, B. J. (2017). The church-turing thesis. In Zalta, E. N., editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, winter 2017 edition.
- Cramér, H. (2016). *Mathematical methods of statistics (PMS-9)*, volume 9. Princeton university press.
- Dalen, van, D. (1997). *Logic and Structure*.
- Darwiche, A. (2010). Bayesian networks. *Commun. ACM*, 53(12):80–90. <http://doi.acm.org/10.1145/1859204.1859227>.
- Darwin, C. (1909). *The Origin of Species*. P. F. Collier & Son. <http://books.google.nl/books?id=YY4EAAAAYAAJ>.
- Davies, P. (2012). Quantum biology: Current status and opportunities. On-line lecture at University of Surrey, UK: <http://www.youtube.com/watch?v=ZKGu1Y8Cxvk>.
- Davis, M., Kumiega, A., and Van Vliet, B. (2013). Ethics, finance, and automation: A preliminary survey of problems in high frequency trading. *Science and Engineering Ethics*, 19(3):851–874.
- De Boor, C. (1978). *A practical guide to splines*, volume 27. Springer-Verlag New York.
- Dempster, A. P. (1968). A generalization of bayesian inference. *Journal of the Royal Statistical Society. Series B (Methodological)*, 30(2):205–247. <http://www.jstor.org/stable/2984504>.
- Dijkstra, E. W. (1979). *On the foolishness of “natural language programming”*, pages 51–53. Springer Berlin Heidelberg, Berlin, Heidelberg. <https://doi.org/10.1007/BFb0014656>.
- Ebbinghaus, H. (1885). Ueber das gedächtnis.
- Edmonds, B., Lucas, P., Rouchier, J., and Taylor, R. (2017). *Human Societies: Understanding Observed Social Phenomena*, pages 763–800. Springer International Publishing, Cham.
- Edmonds, B. and Meyer, R. (2015). *Simulating social complexity*. Springer.
- Edmonds, B. and Meyer, R. (2017). *Simulating social complexity*. Springer, second edition.

- Eiben, A. E. and Smith, J. E. (2003). *Introduction to Evolutionary Computing*. Springer-Verlag.
- Elo, A. E. (1978). *The rating of chessplayers, past and present*. Arco Pub.
- Enderton, H. B. (2015). Second-order and higher-order logic. In Zalta, E. N., editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, fall 2015 edition. <https://plato.stanford.edu/archives/fall2015/entries/logic-higher-order/>.
- Engelbart, D. (1962). Augmenting human intellect: A conceptual framework.
- Engelbart, D. (1995). Toward augmenting the human intellect and boosting our collective iq. *Commun. ACM*, 38(8):30–32.
- Evans, V. (2015). #language: evolution in the digital age. *The Guardian*. <https://www.theguardian.com/media-network/2015/jun/26/hashtag-language-evolution-digital-age>.
- Fillmore, C. J. (1979). On fluency. In *Individual differences in language ability and language behavior*, pages 85–101. Elsevier.
- Fischer, M. J. and Ladner, R. E. (1979). Propositional dynamic logic of regular programs. *Journal of Computer and System Science*, 18(2):194–211.
- Fitch, W. T., Friederici, A. D., and Hagoort, P. (2012). Pattern perception and computational complexity: introduction to the special issue.
- Fitzmaurice, G., Davidian, M., Verbeke, G., and Molenberghs, G. (2008). *Longitudinal data analysis*. CRC press.
- Frege, G. (1879). Begriffsschrift, a formula language, modeled upon that of arithmetic, for pure thought. *From Frege to Gödel: A source book in mathematical logic*, 1931:1–82.
- Friedman, J. H. (1991). Multivariate adaptive regression splines. *The annals of statistics*, pages 1–67.
- Friedrich, R. and Friederici, A. D. (2009). Mathematical logic in the human brain: Syntax. *PLOS ONE*, 4(5):1–7.
- Frischmuth, P., Martin, M., Tramp, S., Riechert, T., and Auer, S. (2015). Ontowiki—an authoring, publication and visualization interface for the data web. *Semantic Web*, 6(3):215–240.

- Fuchs, N. E. and Schwitter, R. (1995). Specifying logic programs in controlled natural language. *arXiv preprint cmp-lg/9507009*.
- Galton, F. (1907). Vox populi (the wisdom of crowds). *Nature*, 75(7):450–451.
- Ganea, M. (2013). On the grammar of first-order logic. *The Romanian Journal of Analytic Philosophy*, 7(1):5–18. <http://www.srfa.ro/rrfa-old/pdf/RRFA-VII-1-Ganea-pp-5-18.pdf>.
- Ganeri, J. (2004). Indian logic. In Gabbay, D. M., Woods, J., and Kanamori, A., editors, *Handbook of the History of Logic*, pages 1–309. Elsevier.
- Gardner, H. (2011). *Frames of mind: The theory of multiple intelligences*. Hachette UK.
- Georgeff, M., Pell, B., Pollack, M., Tambe, M., and Wooldridge, M. (1998). The belief-desire-intention model of agency. In *International Workshop on Agent Theories, Architectures, and Languages*, pages 1–10. Springer.
- Gilbert, N. and Troitzsch, K. (2005). *Simulation for the social scientist*. McGraw-Hill Education (UK).
- Gode, D. K. and Sunder, S. (1993). Allocative efficiency of markets with zero-intelligence traders: Market as a partial substitute for individual rationality. *Journal of political economy*, 101(1):119–137.
- Goertzel, B. (2006). The hidden pattern. *Brown Walker*.
- Goldberg, D. E. and Deb, K. (1991). A comparative analysis of selection schemes used in genetic algorithms. In *Foundations of Genetic Algorithms*, pages 69–93. Morgan Kaufmann.
- Gottwald, S. (2017). Many-valued logic. In Zalta, E. N., editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, winter 2017 edition.
- Gould, S. J. (2002). *The Structure of Evolutionary Theory*. Harvard University Press. <https://books.google.nl/books?id=nhI17e61W0UC>.
- Groenouwe, C. and Top, J. (2008). Real-time translation of thoughts into semantic networks during collective sensemaking: Quality of entity definitions. *Technical Report V20080702*.

- Grossi, D., Gabbay, D., and Van Der Torre, L. (2010). The norm implementation problem in normative multi-agent systems. In *Specification and verification of multi-agent systems*, pages 195–224. Springer.
- Gruber, T. (2009). *Ontology*, pages 1963–1965. Springer US, Boston, MA. https://doi.org/10.1007/978-0-387-39940-9_1318.
- Haarslev, V., Hidde, K., Möller, R., and Wessel, M. (2012). The racerpro knowledge representation and reasoning system. *Semantic Web Journal*, 3(3):267–277.
- Hähnle, R. (2016). Is there a place for user experiments in the ar community? *Newsletter of Association for Automated Reasoning*, (116). <http://www.aarinc.org/Newsletters/116-2016-08.html>.
- Hales, D. (2013). Distributed computer systems. In *Simulating Social Complexity*, pages 563–580. Springer.
- Halstead, M. H. (1977). *Elements of Software Science (Operating and programming systems series)*. Elsevier Science Inc., New York, NY, USA.
- Hancock, P. J. (1994). An empirical comparison of selection methods in evolutionary algorithms. In *AISB Workshop on Evolutionary Computing*, pages 80–94. Springer.
- Harnad, S. (1990). The symbol grounding problem. *Physica D: Nonlinear Phenomena*, 42(1):335 – 346. <http://www.sciencedirect.com/science/article/pii/0167278990900876>.
- Hartmanis, J. and Hopcroft, J. E. (1971). An overview of the theory of computational complexity. *J. ACM*, 18(3):444–475. <http://d.oi.acm.org/10.1145/321650.321661>.
- Hayes, P. (2014). Rdf semantics - w3c recommendation. <https://www.w3.org/TR/2014/REC-rdf11-mt-20140225/>.
- Heylighen, F. (1999). Collective intelligence and its implementation on the web: algorithms to develop a collective mental map. *Computational & Mathematical Organization Theory*, 5(3):253–280.
- Heylighen, F. (2002). The global brain as a new utopia. *Renaissance der Utopie. Zukunftsfiguren des 21. Jahrhunderts*.
- Heylighen, F. (2013). Self-organization in communicating groups: the emergence of coordination, shared references and collective intelligence. In *Complexity Perspectives on Language, Communication and Society*, pages 117–149. Springer.

- Heylighen, F. et al. (2001). The science of self-organization and adaptivity. *The encyclopedia of life support systems*, 5(3):253–280.
- Heylighen, F. and Joslyn, C. (2001). Cybernetics and second-order cybernetics. *Encyclopedia of physical science & technology*, 4:155–170.
- Hilbert, D., Ackermann, W., and Luce, R. (1950). *Principles of Mathematical Logic*. AMS Chelsea Publishing Series. American Mathematical Society. <https://books.google.nl/books?id=7E34DeXok9gC>.
- Hodges, W. (2013). Model theory. <http://plato.stanford.edu/entries/model-theory/>.
- Holliday, R. (1994). Epigenetics: An overview. *Developmental Genetics*, 15(6):453–457. <https://onlinelibrary.wiley.com/doi/abs/10.1002/dvg.1020150602>.
- Kaljurand, K. and Fuchs, N. E. (2007). Verbalizing owl in attempto controlled english. In *OWLED*, volume 258.
- Kennedy, J. (2006). Swarm intelligence. In *Handbook of nature-inspired and innovative computing*, pages 187–219. Springer.
- Klašnja-Milićević, A., Ivanović, M., and Nanopoulos, A. (2015). Recommender systems in e-learning environments: a survey of the state-of-the-art and possible extensions. *Artificial Intelligence Review*, 44(4):571–604. <https://doi.org/10.1007/s10462-015-9440-z>.
- Kline, P. (2013). *Intelligence: The Psychometric View*. Routledge.
- Klinkenberg, S., Straatemeier, M., and van der Maas, H. L. (2011). Computer adaptive practice of maths ability using a new item response model for on the fly ability and difficulty estimation. *Computers & Education*, 57(2):1813–1824.
- Knutov, E., Bra, P. D., and Pechenizkiy, M. (2009). Ah 12 years later: a comprehensive survey of adaptive hypermedia methods and techniques. *New Review of Hypermedia and Multimedia*, 15(1):5–38. <https://doi.org/10.1080/13614560902801608>.
- Kosorukoff, A. (2001). Human based genetic algorithm. In *Systems, Man, and Cybernetics, 2001 IEEE International Conference on*, volume 5, pages 3464–3469. IEEE.
- Kozlowski, S. W. (2012). *The Oxford handbook of organizational psychology*, volume 1. Oxford University Press.

- Krötzsch, M., Simančík, F., and Horrocks, I. (2014). A description logic primer. In Lehmann, J. and Völker, J., editors, *Perspectives on Ontology Learning*, chapter 1. IOS Press.
- Krötzsch, M., Vrandečić, D., and Völkel, M. (2006). Semantic mediawiki. In *International semantic web conference*, pages 935–942. Springer.
- Kruijer, W. (2013). private communication.
- Lange, M. and Lutz, C. (2005). 2-ExpTime lower bounds for propositional dynamic logics with intersection. *Journal of Symbolic Logic*, 70(4):1072–1086.
- Legg, S., Hutter, M., et al. (2007). A collection of definitions of intelligence. *Frontiers in Artificial Intelligence and applications*, 157:17.
- Lehmann, E. L. and Casella, G. (2006). *Theory of point estimation*. Springer Science & Business Media.
- Levy, P. (1999). *Collective Intelligence: Mankind's Emerging World in Cyberspace*. Perseus Publishing.
- Li, W., Calder, R. B., Mar, J. C., and Vijg, J. (2015). Single-cell transcriptogenomics reveals transcriptional exclusion of enu-mutated alleles. *Mutation Research/Fundamental and Molecular Mechanisms of Mutagenesis*, 772:55 – 62.
- Makarios, T. J. M. (2012). A mechanical verification of the independence of tarski's euclidean axiom.
- Malone, T. W. and Bernstein, M. S. (2015). *Handbook of Collective Intelligence*. The MIT Press.
- McGuinness, D. (2004). Owl web ontology language – overview. <http://www.w3.org/TR/owl-features/>.
- Miller, B. L., Goldberg, D. E., et al. (1995). Genetic algorithms, tournament selection, and the effects of noise. *Complex systems*, 9(3):193–212.
- Miller, G. A. (1958). Free recall of redundant strings of letters. *Journal of experimental Psychology*, 56(6):485.
- Mønster, D., Håkonsson, D. D., Eskildsen, J. K., and Wallot, S. (2016). Physiological evidence of interpersonal dynamics in a cooperative production task. *Physiology & behavior*, 156:24–34.

- Moore, D. S. (2015). *The developing genome: An introduction to behavioral epigenetics*. Oxford University Press.
- Morzy, M. (2015). Ict services for open and citizen science. *World Wide Web*, 18(4):1147–1161. <https://doi.org/10.1007/s11280-014-0303-3>.
- Myerson, R. B. (2013). *Game theory*. Harvard university press.
- Nalepa, G. J. (2018). Collaborative knowledge engineering with wikis. In *Modeling with Rules Using Semantic Knowledge Engineering*, pages 355–379. Springer.
- Nash, J. F. et al. (1950). Equilibrium points in n-person games. *Proceedings of the national academy of sciences*, 36(1):48–49.
- Nation, P. (2009). Reading faster. *International Journal of English Studies*, 9(2).
- Neumann, J. v. (1928). Zur theorie der gesellschaftsspiele. *Mathematische annalen*, 100(1):295–320.
- Newell, A. and Simon, H. (1956). The logic theory machine—a complex information processing system. *IRE Transactions on information theory*, 2(3):61–79.
- Ober, J. (2013). Democracy’s wisdom: An aristotelian middle way for collective judgment. *American Political Science Review*, 107(1):104–122.
- Odersky, M., Altherr, P., Cremet, V., Dragos, I., Dubochet, G., Emir, B., McDirmid, S., Micheloud, S., Mihaylov, N., Schinz, M., Spoon, L., Stenman, E., and Zenger, M. (2006). An Overview of the Scala Programming Language (2. edition). Technical report. <http://infoscience.epfl.ch/record/85634/files/ScalaOverview.pdf>.
- Oinas-Kukkonen, H. (2008). Network analysis and crowds of people as sources of new organisational knowledge. *Knowledge Management: Theoretical Foundation*, pages 173–189.
- Osborne, M. J. et al. (2004). *An introduction to game theory*, volume 3. Oxford university press New York.
- Page, S. E. (2010). *Diversity and complexity*. Princeton University Press.
- Paramythis, A. and Loidl-Reisinger, S. (2004). Adaptive learning environments and e-learning standards. *Electronic Journal on e-Learning Volume*, 2(1):181–194.
- Parrington, J. (2017). *The deeper genome: why there is more to the human genome than meets the eye*. Oxford University Press.

- Partee, B. H. (2008). *Compositionality in formal semantics: Selected papers*. John Wiley & Sons.
- Patel-Schneider, P. (2015). private communication.
- Paul Pangaro (2018). Talking about the future of cybernetics. <https://ccsmfa.blog/2018/03/05/talking-about-the-future-of-cybernetics/>. [Online; accessed 15-Jul-2018].
- Pavlik Jr, P. I., Cen, H., and Koedinger, K. R. (2009). Performance factors analysis—a new alternative to knowledge tracing. *Online Submission*.
- Pearce, C. L. and Sims Jr, H. P. (2002). Vertical versus shared leadership as predictors of the effectiveness of change management teams: An examination of aversive, directive, transactional, transformational, and empowering leader behaviors. *Group dynamics: Theory, research, and practice*, 6(2):172.
- Pei, Y. and Takagi, H. (2018). Research progress survey on interactive evolutionary computation. *Journal of Ambient Intelligence and Humanized Computing*, pages 1–14.
- Pelánek, R. (2016). Applications of the elo rating system in adaptive educational systems. *Computers & Education*, 98:169–179.
- Plag, I. (2007). *Introduction to English Linguistics*. Mouton textbook. Mouton de Gruyter. <https://books.google.nl/books?id=0a9No3819R0C>.
- Pothos, E. M. (2007). Theories of artificial grammar learning. *Psychological bulletin*, 133(2):227.
- Pudlák, P. (1998). Chapter viii - the lengths of proofs. In Buss, S. R., editor, *Handbook of Proof Theory*, volume 137 of *Studies in Logic and the Foundations of Mathematics*, pages 547 – 637. Elsevier. <http://www.sciencedirect.com/science/article/pii/S0049237X98800232>.
- Raz, J. (1999). *Practical reason and norms*. OUP Oxford.
- Reber, A. S. (1967). Implicit learning of artificial grammars. *Journal of verbal learning and verbal behavior*, 6(6):855–863. http://www.wjh.harvard.edu/~pal/pdfs/replaced_scanned_articles/reber67_scanned.pdf.
- Riggenbach, H. (1991). Toward an understanding of fluency: A microanalysis of nonnative speaker conversations. *Discourse processes*, 14(4):423–441.

- Rosenberg, L., Baltaxe, D., and Pescetelli, N. (2016). Crowds vs swarms, a comparison of intelligence. In *Swarm/Human Blended Intelligence Workshop (SHBI), 2016*, pages 1–4. IEEE.
- Rosenberg, L. B. (2015). Human swarms, a real-time method for collective intelligence. In *Proceedings of the european conference on artificial life*, pages 658–659.
- Russell, S. J. and Norvig, P. (2016). *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited,.
- Salas, E., Rosen, M. A., Burke, C. S., and Goodwin, G. F. (2009). The wisdom of collectives in organizations: An update of the teamwork competencies. *Team effectiveness in complex organizations: Cross-disciplinary perspectives and approaches*, pages 39–79.
- Schelling, T. C. (1971). Dynamic models of segregation. *Journal of mathematical sociology*, 1(2):143–186.
- Schild, K. (1989). Towards a theory of frames and rules. KIT-Report 76, Fachbereich Informatik, Technische Universität Berlin, Berlin, Germany.
- Schoenauer, M. and Michalewicz, Z. (1997). Evolutionary computation.
- Schoenberg, I. (1946). Contributions to the problem of approximation of equidistant data by analytic functions: Part a.—on the problem of smoothing or graduation. a first class of analytic approximation formulae. *Quarterly of Applied Mathematics*, 4(1):45–99.
- Schreiber, G. (2002). The web is not well-formed. *IEEE Intelligent Systems*, 17(2):79–80. <http://www.cs.vu.nl/~guus/papers/Schreiber02a.pdf>.
- Schulz, S. (2013). System Description: E 1.8. In McMillan, K., Middeldorp, A., and Voronkov, A., editors, *Proc. of the 19th LPAR, Stellenbosch*, volume 8312 of *LNCS*. Springer.
- Selevičienė, E. and Burkšaitienė, N. (2016). Cmaptools and its use in education. *Journal of Teaching English for Specific and Academic Purposes*, 4(3):631–640. https://www.researchgate.net/profile/Nijole_Burksaitiene/publication/313799621_CmapTools_and_its_use_in_education/links/58a6db2aa6fdcc0e078aac4c/CmapTools-and-its-use-in-education.pdf.
- Sen, A. and Srivastava, M. (2012). *Regression analysis: theory, methods, and applications*. Springer Science & Business Media.

- Shafer, G. (1976). *A mathematical theory of evidence*, volume 42. Princeton university press.
- Siegmund, J., Kästner, C., Apel, S., Parnin, C., Bethmann, A., Leich, T., Saake, G., and Brechmann, A. (2014). Understanding understanding source code with functional magnetic resonance imaging. In *Proceedings of the 36th International Conference on Software Engineering*, pages 378–389. ACM.
- Simon, D. (2013). *Evolutionary Optimization Algorithms*.
- Singh, A. and Haahr, M. (2006). Creating an adaptive network of hubs using schelling's model. *Communications of the ACM*, 49(3):69–73.
- Siorpaes, K. and Hepp, M. (2008). Games with a purpose for the semantic web. *IEEE Intelligent Systems*, 23(3):50–60.
- Smith, R. (2015). Aristotle's logic. In Zalta, E. N., editor, *The Stanford Encyclopedia of Philosophy*. Summer 2015 edition. <http://plato.stanford.edu/archives/sum2015/entries/aristotle-logic/>.
- Sowa, J. (1992). Semantic networks. *Encyclopedia of Artificial Intelligence*. <http://www.jfsowa.com/pubs/semnet.htm>.
- Spearman, C. (1904). "general intelligence," objectively determined and measured. *The American Journal of Psychology*, 15(2):201–292.
- Squazzoni, F. (2008). The micro-macro link in social simulation. *Sociologica*, 2(1):0–0.
- Stahl, G., Koschmann, T. D., and Suthers, D. D. (2006). *Computer-supported collaborative learning*. na.
- Sternberg, R. (2017). intelligence. <https://www.britannica.com/science/human-intelligence-psychology>, the version with latest update at April 26, 2017.
- Surowiecki, J. (2005). *The wisdom of crowds*. Anchor.
- Sutcliffe, G. (2010). The tptp world – infrastructure for automated reasoning. In Clarke, E. M. and Voronkov, A., editors, *Logic for Programming, Artificial Intelligence, and Reasoning*, pages 1–12, Berlin, Heidelberg. Springer Berlin Heidelberg.
- Sutcliffe, G. (2017). The TPTP Problem Library and Associated Infrastructure. From CNF to TH0, TPTP v6.4.0. *Journal of Automated Reasoning*, 59(4):483–502.

- Szabó, Z. G. (2017). Compositionality. In Zalta, E. N., editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, summer 2017 edition.
- Takagi, H. (2001). Interactive evolutionary computation: Fusion of the capabilities of ec optimization and human evaluation. *Proceedings of the IEEE*, 89(9):1275–1296.
- Thierens, D. (2016). private communication.
- Toppino, T. C. and Gerbier, E. (2014). About practice: Repetition, spacing, and abstraction. In *Psychology of learning and motivation*, volume 60, pages 113–189. Elsevier.
- Turing, A. M. (1936). On computable numbers, with an application to the Entscheidungsproblem. *Mathematical Proceedings of the Cambridge Philosophical Society*, 42:230–265. https://www.cs.virginia.edu/~robins/Turing_Paper_1936.pdf.
- Venners, B. and Sommers, F. (2009). The origins of scala. https://www.artima.com/scalazine/articles/origins_of_scala.html, retrieved January 28th 2019.
- Vincent, D. (2000). *The rise of mass literacy: Reading and writing in modern Europe*. Blackwell Publishing.
- Visser, J. (2001). Learning communities: Wholeness and partness, autonomy and dependence in the learning ecology. <http://www.learndev.org/dl/Barcelona2001.pdf>.
- Visser, J. and Berg, D. (1999). Learning without frontiers: Building integrated responses to diverse learning needs. <http://www.learndev.org/dl/aect99-etrtd.pdf>.
- Vrandečić, D. and Krötzsch, M. (2014). Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85.
- W3C (2018). Introductory webpage about the semantic web. *W3C website*. <https://www.w3.org/standards/semanticweb/> retrieved March 11, 2018.
- W3C OWL Working Group (2012). Owl 2 web ontology language document overview (second edition). <https://www.w3.org/TR/2012/REC-owl2-overview-20121211/>.

- W3C SPARQL Working Group (Version of March 21th, 2013). Turtle - terse rdf triple language. <https://www.w3.org/TR/2013/REC-sparql11-overview-20130321>.
- Wald, A. (1947). *Sequential Analysis*. John Wiley and Sons, 1st edition. <https://books.google.nl/books?id=oVYDHHzZtdIC>.
- Watson, J. D., Crick, F. H., et al. (1953). Molecular structure of nucleic acids. *Nature*, 171(4356):737–738.
- Westerståhl, D. (2016). Generalized quantifiers. In Zalta, E. N., editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, winter 2016 edition. <https://plato.stanford.edu/archives/win2016/entries/generalized-quantifiers/>.
- Whitley, L. D. (1989). The genitor algorithm and selection pressure: Why rank-based allocation of reproductive trials is best. In *ICGA*, volume 89, pages 116–123. Fairfax, VA.
- Wiedijk, F. (2015). Formalizing 100 theorems. <http://www.cs.ru.nl/~freek/100/index.html>.
- Wiener, N. (1961). *Cybernetics or Control and Communication in the Animal and the Machine*, volume 25. MIT press.
- Wiggins, A. and Crowston, K. (2011). From conservation to crowdsourcing: A typology of citizen science. In *System Sciences (HICSS), 2011 44th Hawaii international conference on*, pages 1–10. IEEE.
- Wikipedia (2015). First-order logic. https://en.wikipedia.org/wiki/First-order_logic, retrieved July 19th, 2015).
- Wikipedia (2018a). Power set. https://en.wikipedia.org/w/index.php?title=Power_set&oldid=875446448, version of 16:57, December 26th, 2018.
- Wikipedia (2018c). Semantic web stack. https://en.wikipedia.org/w/index.php?title=Semantic_Web_Stack&oldid=681457479, retrieved March 11, 2018.
- Wikipedia (2018d). Social simulation — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Social_simulation&oldid=847333510, retrieved July 20th, 2018.

- Wikipedia (2018e). Time series. https://en.wikipedia.org/w/index.php?title=Time_series&oldid=838494271, version 11:29, April 17th, 2018.
- Wikipedia (Version of 10:28, March 13th, 2018b). Propositional calculus. https://en.wikipedia.org/w/index.php?title=Propositional_calculus&oldid=830196187.
- Wikipedia contributors (2018). Elo rating system — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Elo_rating_system&oldid=841341186, retrieved May 27th 2018.
- Wikipedia contributors (2019). Step function — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Step_function&oldid=898281551, retrieved July the 2nd 2019.
- Wiles, A. (1995). Modular elliptic curves and fermat's last theorem. *Annals of Mathematics*, 141(3):443–551. <http://www.jstor.org/stable/2118559>.
- Wilson, P. (1991). *Computer supported cooperative work:: An introduction*. Springer Science & Business Media.
- Wood, P. J. (2011). Climate change and game theory. *Annals of the New York Academy of Sciences*, 1219(1):153–170.
- Wooldridge, M. (2009). *An introduction to multiagent systems*. John Wiley & Sons.
- Woolley, A. W., Chabris, C. F., Pentland, A., Hashmi, N., and Malone, T. W. (2010). Evidence for a collective intelligence factor in the performance of human groups. *science*, 330(6004):686–688.
- Yukl, G. A. (2013). *Leadership in organizations*. Pearson Education India, 8th edition.
- Zadeh, L. A. (1988). Fuzzy logic. *Computer*, 21(4):83–93.

SIKS dissertation series

-
- 2011 01 Botond Cseke (RUN), Variational Algorithms for Bayesian Inference in Latent Gaussian Models
02 Nick Tinnemeier (UU), Organizing Agent Organizations. Syntax and Operational Semantics of an Organization-Oriented Programming Language
03 Jan Martijn van der Werf (TUE), Compositional Design and Verification of Component-Based Information Systems
04 Hado van Hasselt (UU), Insights in Reinforcement Learning: Formal analysis and empirical evaluation of temporal-difference
05 Bas van der Raadt (VU), Enterprise Architecture Coming of Age - Increasing the Performance of an Emerging Discipline.
06 Yiwen Wang (TUE), Semantically-Enhanced Recommendations in Cultural Heritage
07 Yujia Cao (UT), Multimodal Information Presentation for High Load Human Computer Interaction
08 Nieske Vergunst (UU), BDI-based Generation of Robust Task-Oriented Dialogues
09 Tim de Jong (OU), Contextualised Mobile Media for Learning
10 Bart Bogaert (UvT), Cloud Content Contention
11 Dhaval Vyas (UT), Designing for Awareness: An Experience-focused HCI Perspective
12 Carmen Bratosin (TUE), Grid Architecture for Distributed Process Mining
13 Xiaoyu Mao (UvT), Airport under Control. Multiagent Scheduling for Airport Ground Handling
14 Milan Lovric (EUR), Behavioral Finance and Agent-Based Artificial Markets
15 Marijn Koolen (UvA), The Meaning of Structure: the Value of Link Evidence for Information Retrieval
16 Maarten Schadd (UM), Selective Search in Games of Different Complexity
17 Jiyin He (UVA), Exploring Topic Structure: Coherence, Diversity and Relatedness
18 Mark Ponsen (UM), Strategic Decision-Making in complex games
19 Ellen Rusman (OU), The Mind's Eye on Personal Profiles
20 Qing Gu (VU), Guiding service-oriented software engineering - A view-based approach
21 Linda Terlouw (TUD), Modularization and Specification of Service-Oriented Systems
22 Junte Zhang (UVA), System Evaluation of Archival Description and Access
23 Wouter Weerkamp (UVA), Finding People and their Utterances in Social Media
24 Herwin van Welbergen (UT), Behavior Generation for Interpersonal Coordination with Virtual Humans On Specifying, Scheduling and Realizing Multi-modal Virtual Human Behavior
25 Syed Waqar ul Qounain Jaffry (VU), Analysis and Validation of Models for Trust Dynamics
26 Matthijs Aart Pontier (VU), Virtual Agents for Human Communication - Emotion Regulation and Involvement-Distance Trade-Offs in Embodied Conversational Agents and Robots
27 Aniel Bhulai (VU), Dynamic website optimization through autonomous management of design patterns
28 Rianne Kaptein (UVA), Effective Focused Retrieval by Exploiting Query Context and Document Structure
29 Faisal Kamiran (TUE), Discrimination-aware Classification
30 Egon van den Broek (UT), Affective Signal Processing (ASP): Unraveling the mystery of emotions
31 Ludo Waltman (EUR), Computational and Game-Theoretic Approaches for Modeling Bounded Rationality
32 Nees-Jan van Eck (EUR), Methodological Advances in Bibliometric Mapping of Science
33 Tom van der Weide (UU), Arguing to Motivate Decisions
34 Paolo Turrini (UU), Strategic Reasoning in Interdependence: Logical and Game-theoretical Investigations
35 Maaikje Harbers (UU), Explaining Agent Behavior in Virtual Training
36 Erik van der Spek (UU), Experiments in serious game design: a cognitive approach
37 Adriana Burlutiu (RUN), Machine Learning for Pairwise Data, Applications for Preference Learning and Supervised Network Inference
38 Nyree Lemmens (UM), Bee-inspired Distributed Optimization
39 Joost Westra (UU), Organizing Adaptation using Agents in Serious Games
40 Viktor Clerc (VU), Architectural Knowledge Management in Global Software Development
41 Luan Ibraimi (UT), Cryptographically Enforced Distributed Data Access Control
42 Michal Sindlar (UU), Explaining Behavior through Mental State Attribution
43 Henk van der Schuur (UU), Process Improvement through Software Operation Knowledge
44 Boris Reuderink (UT), Robust Brain-Computer Interfaces
45 Herman Stehouwer (UvT), Statistical Language Models for Alternative Sequence Selection
46 BeiBei Hu (TUD), Towards Contextualized Information Delivery: A Rule-based Architecture for the Domain of Mobile Police Work
47 Azizi Bin Ab Aziz (VU), Exploring Computational Models for Intelligent Support of Persons with Depression
48 Mark Ter Maat (UT), Response Selection and Turn-taking for a Sensitive Artificial Listening Agent
49 Andreea Niculescu (UT), Conversational interfaces for task-oriented spoken dialogues: design aspects influencing interaction quality
-
- 2012 01 Terry Kakeeto (UvT), Relationship Marketing for SMEs in Uganda
02 Muhammad Unair (VU), Adaptivity, emotion, and Rationality in Human and Ambient Agent Models
03 Adam Vanya (VU), Supporting Architecture Evolution by Mining Software Repositories
04 Jurriaan Souer (UU), Development of Content Management System-based Web Applications

- 05 Marijn Plomp (UU), Maturing Interorganisational Information Systems
06 Wolfgang Reinhardt (OU), Awareness Support for Knowledge Workers in Research Networks
07 Rianne van Lambalgen (VU), When the Going Gets Tough: Exploring Agent-based Models of Human Performance under Demanding Conditions
08 Gerben de Vries (UVA), Kernel Methods for Vessel Trajectories
09 Ricardo Neisse (UT), Trust and Privacy Management Support for Context-Aware Service Platforms
10 David Smits (TUE), Towards a Generic Distributed Adaptive Hypermedia Environment
11 J.C.B. Ranthan Prabhakara (TUE), Process Mining in the Large: Preprocessing, Discovery, and Diagnostics
12 Kees van der Sluijs (TUE), Model Driven Design and Data Integration in Semantic Web Information Systems
13 Saleman Shahid (UVT), Fun and Face: Exploring non-verbal expressions of emotion during playful interactions
14 Evgeny Knutov (TUE), Generic Adaptation Framework for Unifying Adaptive Web-based Systems
15 Natalie van der Wal (VU), Social Agents. Agent-Based Modelling of Integrated Internal and Social Dynamics of Cognitive and Affective Processes.
16 Fienke Both (VU), Helping people by understanding them - Ambient Agents supporting task execution and depression treatment
17 Amal Elgammal (UVT), Towards a Comprehensive Framework for Business Process Compliance
18 Eltjo Poort (VU), Improving Solution Architecting Practices
19 Helen Schonenberg (TUE), What's Next? Operational Support for Business Process Execution
20 Ali Bahramisharif (RUN), Covert Visual Spatial Attention, a Robust Paradigm for Brain-Computer Interfacing
21 Roberto Cornacchia (TUD), Querying Sparse Matrices for Information Retrieval
22 Thijs Vis (UvT), Intelligence, politie en veiligheidsdienst: verenigbare grootheden?
23 Christian Muehl (UT), Toward Affective Brain-Computer Interfaces: Exploring the Neurophysiology of Affect during Human Media Interaction
24 Laurens van der Werff (UT), Evaluation of Noisy Transcripts for Spoken Document Retrieval
25 Silja Eckartz (UT), Managing the Business Case Development in Inter-Organizational IT Projects: A Methodology and its Application
26 Emile de Maat (UVA), Making Sense of Legal Text
27 Hayretin Gurkok (UT), Mind the Sheep! User Experience Evaluation & Brain-Computer Interface Games
28 Nancy Pascall (UvT), Engendering Technology Empowering Women
29 Almer Tigelaar (UT), Peer-to-Peer Information Retrieval
30 Alina Pommeranz (TUD), Designing Human-Centered Systems for Reflective Decision Making
31 Emily Bagarukayo (RUN), A Learning by Construction Approach for Higher Order Cognitive Skills Improvement, Building Capacity and Infrastructure
32 Wietske Visser (TUD), Qualitative multi-criteria preference representation and reasoning
33 Rory Sie (OUN), Coalitions in Cooperation Networks (COCOON)
34 Pavol Jancura (RUN), Evolutionary analysis in PPI networks and applications
35 Evert Haasdijs (VU), Never Too Old To Learn - On-line Evolution of Controllers in Swarm- and Modular Robotics
36 Denis Szebunawo (RUN), Analysis and Evaluation of Collaborative Modeling Processes
37 Agnes Nakakawa (RUN), A Collaboration Process for Enterprise Architecture Creation
38 Selmar Smit (VU), Parameter Tuning and Scientific Testing in Evolutionary Algorithms
39 Hassan Fatemi (UT), Risk-aware design of value and coordination networks
40 Agus Gunawan (UvT), Information Access for SMEs in Indonesia
41 Sebastian Kelle (OU), Game Design Patterns for Learning
42 Dominique Verpoorten (OU), Reflection Amplifiers in self-regulated Learning
43 Withdrawn
44 Anna Tordai (VU), On Combining Alignment Techniques
45 Benedikt Kratz (UvT), A Model and Language for Business-aware Transactions
46 Simon Carter (UVA), Exploration and Exploitation of Multilingual Data for Statistical Machine Translation
47 Manos Tsagkias (UVA), Mining Social Media: Tracking Content and Predicting Behavior
48 Jörn Bakker (TUE), Handling Abrupt Changes in Evolving Time-series Data
49 Michael Kaisers (UM), Learning against Learning - Evolutionary dynamics of reinforcement learning algorithms in strategic interactions
50 Steven van Kervel (TUD), Ontology driven Enterprise Information Systems Engineering
51 Jeroen de Jong (TUD), Heuristics in Dynamic Scheduling; a practical framework with a case study in elevator dispatching
-
- 2013 01 Viorel Milea (EUR), News Analytics for Financial Decision Support
02 Erietta Liarou (CWI), MonetDB/DataCell: Leveraging the Column-store Database Technology for Efficient and Scalable Stream Processing
03 Szymon Klarman (VU), Reasoning with Contexts in Description Logics
04 Chetan Yadati (TUD), Coordinating autonomous planning and scheduling
05 Dulce Pumaraja (UT), Groupware Requirements Evolutions Patterns
06 Romulo Goncalves (CWI), The Data Cyclotron: Juggling Data and Queries for a Data Warehouse Audience
07 Giel van Lankveld (UvT), Quantifying Individual Player Differences
08 Robbert-Jan Merk (VU), Making enemies: cognitive modeling for opponent agents in fighter pilot simulators
09 Fabio Gori (RUN), Metagenomic Data Analysis: Computational Methods and Applications
10 Jeevanie Jayasinghe Arachchige (UvT), A Unified Modeling Framework for Service Design.
11 Evangelos Pourmaras (TUD), Multi-level Reconfigurable Self-organization in Overlay Services
12 Marian Razavian (VU), Knowledge-driven Migration to Services
13 Mohammad Safiri (UT), Service Tailoring: User-centric creation of integrated IT-based homecare services to support independent living of elderly
14 Jafar Tanha (UVA), Ensemble Approaches to Semi-Supervised Learning
15 Daniel Hennes (UM), Multiagent Learning - Dynamic Games and Applications
16 Eric Kok (UU), Exploring the practical benefits of argumentation in multi-agent deliberation
17 Koen Kok (VU), The PowerMatcher: Smart Coordination for the Smart Electricity Grid
18 Jeroen Janssens (UvT), Outlier Selection and One-Class Classification
19 Renze Steenhuisen (TUD), Coordinated Multi-Agent Planning and Scheduling
20 Katja Hofmann (UvA), Fast and Reliable Online Learning to Rank for Information Retrieval
21 Sander Wubben (UvT), Text-to-text generation by monolingual machine translation
22 Tom Claassen (RUN), Causal Discovery and Logic
23 Patrício de Alencar Silva (UvT), Value Activity Monitoring
24 Hailham Bou Ammar (UM), Automated Transfer in Reinforcement Learning
25 Agnieszka Anna Latoszek-Berendsen (UM), Intention-based Decision Support. A new way of representing and implementing clinical guidelines in a Decision Support System
26 Alireza Zarghami (UT), Architectural Support for Dynamic Homecare Service Provisioning
27 Mohammad Huq (UT), Inference-based Framework Managing Data Provenance
28 Frans van der Sluis (UT), When Complexity becomes Interesting: An Inquiry into the Information eXperience
29 Iwan de Kok (UT), Listening Heads
30 Joyce Nakatumba (TUE), Resource-Aware Business Process Management: Analysis and Support

-
- 31 Dinh Khoa Nguyen (UvT), Blueprint Model and Language for Engineering Cloud Applications
32 Kamakshi Rajagopal (OUN), Networking For Learning: The role of Networking in a Lifelong Learner's Professional Development
33 Qi Gao (TUD), User Modeling and Personalization in the Microblogging Sphere
34 Kien Tjin-Kam-Jet (UT), Distributed Deep Web Search
35 Abdallah El Ali (UvA), Minimal Mobile Human Computer Interaction
36 Than Lam Hoang (TUE), Pattern Mining in Data Streams
37 Dirk Börner (OUN), Ambient Learning Displays
38 Eelco den Heijer (VU), Autonomous Evolutionary Art
39 Joop de Jong (TUD), A Method for Enterprise Ontology based Design of Enterprise Information Systems
40 Pim Nijssen (UM), Monte-Carlo Tree Search for Multi-Player Games
41 Jochem Liem (UVA), Supporting the Conceptual Modelling of Dynamic Systems: A Knowledge Engineering Perspective on Qualitative Reasoning
42 Léon Planken (TUD), Algorithms for Simple Temporal Reasoning
43 Marc Bron (UVA), Exploration and Contextualization through Interaction and Concepts
-
- 2014 01 Nicola Barile (UU), Studies in Learning Monotone Models from Data
02 Fiona Tullyano (RUN), Combining System Dynamics with a Domain Modeling Method
03 Sergio Raul Duarte Torres (UT), Information Retrieval for Children: Search Behavior and Solutions
04 Hanna Jochmann-Mannak (UT), Websites for children: search strategies and interface design - Three studies on children's search performance and evaluation
05 Jurriaan van Reijns (UU), Knowledge Perspectives on Advancing Dynamic Capability
06 Damian Tamburri (VU), Supporting Networked Software Development
07 Arya Adriansyah (TUE), Aligning Observed and Modeled Behavior
08 Samur Araujo (TUD), Data Integration over Distributed and Heterogeneous Data Endpoints
09 Philip Jackson (UvT), Toward Human-Level Artificial Intelligence: Representation and Computation of Meaning in Natural Language
10 Ivan Salvador Razo Zapata (VU), Service Value Networks
11 Janneke van der Zwaan (TUD), An Empathic Virtual Buddy for Social Support
12 Willem van Willigen (VU), Look Ma, No Hands: Aspects of Autonomous Vehicle Control
13 Arlette van Wissen (VU), Agent-Based Support for Behavior Change: Models and Applications in Health and Safety Domains
14 Yangyang Shi (TUD), Language Models With Meta-information
15 Natalya Mogles (VU), Agent-Based Analysis and Support of Human Functioning in Complex Socio-Technical Systems: Applications in Safety and Healthcare
16 Krystyna Milian (VU), Supporting trial recruitment and design by automatically interpreting eligibility criteria
17 Kathrin Dentler (VU), Computing healthcare quality indicators automatically: Secondary Use of Patient Data and Semantic Interoperability
18 Mattijs Ghijzen (UVA), Methods and Models for the Design and Study of Dynamic Agent Organizations
19 Vinicius Ramos (TUE), Adaptive Hypermedia Courses: Qualitative and Quantitative Evaluation and Tool Support
20 Mena Habib (UT), Named Entity Extraction and Disambiguation for Informal Text: The Missing Link
21 Cassidy Clark (TUD), Negotiation and Monitoring in Open Environments
22 Marieke Peeters (UU), Personalized Educational Games - Developing agent-supported scenario-based training
23 Eleftherios Sidirouros (Uva/CWI), Space Efficient Indexes for the Big Data Era
24 Davide Ceolin (VU), Trusting Semi-structured Web Data
25 Martijn Lappenschaar (RUN), New network models for the analysis of disease interaction
26 Tim Baarslag (TUD), What to Bid and When to Stop
27 Rui Jorge Almeida (EUR), Conditional Density Models Integrating Fuzzy and Probabilistic Representations of Uncertainty
28 Anna Chmielowiec (VU), Decentralized k-Clique Matching
29 Jaap Kabbedijk (UU), Variability in Multi-Tenant Enterprise Software
30 Peter de Cock (UvT), Anticipating Criminal Behaviour
31 Leo van Moergestel (UU), Agent Technology in Agile Multiparallel Manufacturing and Product Support
32 Naser Ayat (UvA), On Entity Resolution in Probabilistic Data
33 Tesfa Tegegne (RUN), Service Discovery in eHealth
34 Christina Manteli (VU), The Effect of Governance in Global Software Development: Analyzing Transactive Memory Systems.
35 Joost van Ooijen (UU), Cognitive Agents in Virtual Worlds: A Middleware Design Approach
36 Joos Buijs (TUE), Flexible Evolutionary Algorithms for Mining Structured Process Models
37 Maral Dadvar (UT), Experts and Machines United Against Cyberbullying
38 Danny Plass-Oude Bos (UT), Making brain-computer interfaces better: improving usability through post-processing.
39 Jasmina Maric (UvT), Web Communities, Immigration, and Social Capital
40 Walter Omona (RUN), A Framework for Knowledge Management Using ICT in Higher Education
41 Frederic Hogenboom (EUR), Automated Detection of Financial Events in News Text
42 Carsten Eijckhof (CWI/TUD), Contextual Multidimensional Relevance Models
43 Kevin Vlaanderen (UU), Supporting Process Improvement using Method Increments
44 Paulien Meesters (UvT), Intelligent Blauw. Met als ondertitel: Intelligence-gestuurde politiezorg in gebiedsgebonden eenheden.
45 Birgit Schmitz (OUN), Mobile Games for Learning: A Pattern-Based Approach
46 Ke Tao (TUD), Social Web Data Analytics: Relevance, Redundancy, Diversity
47 Shangsong Liang (UVA), Fusion and Diversification in Information Retrieval
-
- 2015 01 Niels Netten (UvA), Machine Learning for Relevance of Information in Crisis Response
02 Faiza Bukhsh (UvT), Smart auditing: Innovative Compliance Checking in Customs Controls
03 Twan van Laarhoven (RUN), Machine learning for network data
04 Howard Spoelstra (OUN), Collaborations in Open Learning Environments
05 Christoph Bösch (UT), Cryptographically Enforced Search Pattern Hiding
06 Farideh Heidari (TUD), Business Process Quality Computation - Computing Non-Functional Requirements to Improve Business Processes
07 Maria-Hendrike Peetz (UvA), Time-Aware Online Reputation Analysis
08 Jie Jiang (TUD), Organizational Compliance: An agent-based model for designing and evaluating organizational interactions
09 Randy Klaassen (UT), HCI Perspectives on Behavior Change Support Systems
10 Henry Hermans (OUN), OpenU: design of an integrated system to support lifelong learning
11 Yongming Luo (TUE), Designing algorithms for big graph datasets: A study of computing bisimulation and joins
12 Julie M. Birkholz (VU), Modi Operandi of Social Network Dynamics: The Effect of Context on Scientific Collaboration Networks
13 Giuseppe Procaccianti (VU), Energy-Efficient Software
14 Bart van Straalen (UT), A cognitive approach to modeling bad news conversations
15 Klaas Andries de Graaf (VU), Ontology-based Software Architecture Documentation

- 16 Changyun Wei (UT), Cognitive Coordination for Cooperative Multi-Robot Teamwork
 17 André van Cleef (UT), Physical and Digital Security Mechanisms: Properties, Combinations and Trade-offs
 18 Holger Pirk (CWI), Waste Not, Want Not! - Managing Relational Data in Asymmetric Memories
 19 Bernardo Tabuenca (OUN), Ubiquitous Technology for Lifelong Learners
 20 Lois Vanhée (UU), Using Culture and Values to Support Flexible Coordination
 21 Sibren Fetter (OUN), Using Peer-Support to Expand and Stabilize Online Learning
 22 Zhemín Zhu (UT), Co-occurrence Rate Networks
 23 Luit Gazendam (VU), Cataloguer Support in Cultural Heritage
 24 Richard Berendsen (UVA), Finding People, Papers, and Posts: Vertical Search Algorithms and Evaluation
 25 Steven Woudenberg (UU), Bayesian Tools for Early Disease Detection
 26 Alexander Hogenboom (EUR), Sentiment Analysis of Text Guided by Semantics and Structure
 27 Sándor Héman (CWI), Updating compressed column stores
 28 Janet Bagorogoza (TU), Knowledge Management and High Performance: The Uganda Financial Institutions Model for HPO
 29 Hendrik Baier (UM), Monte-Carlo Tree Search Enhancements for One-Player and Two-Player Domains
 30 Kiavash Bahreini (OU), Real-time Multimodal Emotion Recognition in E-Learning
 31 Yakup Koç (TUD), On the robustness of Power Grids
 32 Jerome Gard (UL), Corporate Venture Management in SMEs
 33 Frederik Schadd (TUD), Ontology Mapping with Auxiliary Resources
 34 Victor de Graaf (UT), Gesocial Recommender Systems
 35 Jungxiao Xu (TUD), Affective Body Language of Humanoid Robots: Perception and Effects in Human Robot Interaction
-
- 2016 01 Syed Saïden Abbas (RUN), Recognition of Shapes by Humans and Machines
 02 Michiel Christiaan Meulendijk (UU), Optimizing medication reviews through decision support: prescribing a better pill to swallow
 03 Maya Sappelli (RUN), Knowledge Work in Context: User Centered Knowledge Worker Support
 04 Laurens Rietveld (VU), Publishing and Consuming Linked Data
 05 Evgeny Sherkhonov (UVA), Expanded Acyclic Queries: Containment and an Application in Explaining Missing Answers
 06 Michel Wilson (TUD), Robot scheduling in an uncertain environment
 07 Jeroen de Man (VU), Measuring and modeling negative emotions for virtual training
 08 Matje van de Camp (TU), A Link to the Past: Constructing Historical Social Networks from Unstructured Data
 09 Archana Nottankandath (VU), Trusting Crowdsourced Information on Cultural Artefacts
 10 George Karafotias (VUA), Parameter Control for Evolutionary Algorithms
 11 Anne Schuth (UVA), Search Engines that Learn from Their Users
 12 Max Knobout (UU), Logics for Modelling and Verifying Normative Multi-Agent Systems
 13 Nana Baah Gyan (VU), The Web, Speech Technologies and Rural Development in West Africa - An ICT4D Approach
 14 Ravi Khadka (UU), Revisiting Legacy Software System Modernization
 15 Steffen Michels (RUN), Hybrid Probabilistic Logics - Theoretical Aspects, Algorithms and Experiments
 16 Guangliang Li (UVA), Socially Intelligent Autonomous Agents that Learn from Human Reward
 17 Berend Weel (VU), Towards Embodied Evolution of Robot Organisms
 18 Albert Merotto Peñuela (VU), Refining Statistical Data on the Web
 19 Julia Efremova (Tue), Mining Social Structures from Genealogical Data
 20 Daan Odijk (UVA), Context & Semantics in News & Web Search
 21 Alejandro Moreno Celleri (UT), From Traditional to Interactive Playspaces: Automatic Analysis of Player Behavior in the Interactive Tag Playground
 22 Grace Lewis (VU), Software Architecture Strategies for Cyber-Foraging Systems
 23 Fei Cai (UVA), Query Auto Completion in Information Retrieval
 24 Brend Wanders (UT), Repurposing and Probabilistic Integration of Data: An Iterative and data model independent approach
 25 Julia Kiseleva (Tue), Using Contextual Information to Understand Searching and Browsing Behavior
 26 Dilhan Thilakarathne (VU), In or Out of Control: Exploring Computational Models to Study the Role of Human Awareness and Control in Behavioural Choices, with Applications in Aviation and Energy Management Domains
 27 Wen Li (TUD), Understanding Geo-spatial Information on Social Media
 28 Mingxin Zhang (TUD), Large-scale Agent-based Social Simulation - A study on epidemic prediction and control
 29 Nicolas Höning (TUD), Peak reduction in decentralised electricity systems - Markets and prices for flexible planning
 30 Ruid Mattheij (UvT), The Eyes Have It
 31 Mohammad Khelghati (UT), Deep web content monitoring
 32 Elco Vriezekolk (UT), Assessing Telecommunication Service Availability Risks for Crisis Organisations
 33 Peter Bloem (UVA), Single Sample Statistics, exercises in learning from just one example
 34 Dennis Schunselaar (TUE), Configurable Process Trees: Elicitation, Analysis, and Enactment
 35 Zhaochun Ren (UVA), Monitoring Social Media: Summarization, Classification and Recommendation
 36 Daphne Karreman (UT), Beyond R2D2: The design of nonverbal interaction behavior optimized for robot-specific morphologies
 37 Giovanni Sileno (UvA), Aligning Law and Action - a conceptual and computational inquiry
 38 Andrea Minuto (UT), Materials that Matter - Smart Materials meet Art & Interaction Design
 39 Merijn Bruijns (UT), Believable Suspect Agents: Response and Interpersonal Style Selection for an Artificial Suspect
 40 Christian Detweiler (TUD), Accounting for Values in Design
 41 Thomas King (TUD), Governing Governance: A Formal Framework for Analysing Institutional Design and Enactment Governance
 42 Spyros Martzoukos (UVA), Combinatorial and Compositional Aspects of Bilingual Aligned Corpora
 43 Saskia Koldijk (RUN), Context-Aware Support for Stress Self-Management: From Theory to Practice
 44 Thibault Sellam (UVA), Automatic Assistants for Database Exploration
 45 Bram van de Laar (UT), Experiencing Brain-Computer Interface Control
 46 Jorge Gallego Perez (UT), Robots to Make you Happy
 47 Christina Weber (UL), Real-time foresight - Preparedness for dynamic innovation networks
 48 Tanja Buttler (TUD), Collecting Lessons Learned
 49 Gleb Polevoy (TUD), Participation and Interaction in Projects. A Game-Theoretic Analysis
 50 Yan Wang (UvT), The Bridge of Dreams: Towards a Method for Operational Performance Alignment in IT-enabled Service Supply Chains
-
- 2017 01 Jan-Jaap Oerlemans (UL), Investigating Cybercrime
 02 Sjoerd Timmer (UU), Designing and Understanding Forensic Bayesian Networks using Argumentation
 03 Daniël Harold Telgen (UU), Grid Manufacturing: A Cyber-Physical Approach with Autonomous Products and Reconfigurable Manufacturing Machines
 04 Mrunal Gawade (CWI), Multi-core Parallelism in a Column-store
 05 Mahdieh Shadi (UVA), Collaboration Behavior
 06 Damir Vandić (EUR), Intelligent Information Systems for Web Product Search

- 07 Roel Bertens (UU), Insight in Information: from Abstract to Anomaly
08 Rob Konijn (VU), Detecting Interesting Differences: Data Mining in Health Insurance Data using Outlier Detection and Subgroup Discovery
09 Dong Nguyen (UT), Text as Social and Cultural Data: A Computational Perspective on Variation in Text
10 Robby van Delden (UT), (Steering) Interactive Play Behavior
11 Florian Kuneman (RUN), Modelling patterns of time and emotion in Twitter #anticipinment
12 Sander Leemans (TUE), Robust Process Mining with Guarantees
13 Gijs Huisman (UT), Social Touch Technology - Extending the reach of social touch through haptic technology
14 Shoshannah Tekofsky (UVT), You Are Who You Play You Are: Modelling Player Traits from Video Game Behavior
15 Peter Berck (RUN), Memory-Based Text Correction
16 Aleksandr Chuklin (UVA), Understanding and Modeling Users of Modern Search Engines
17 Daniel Dimov (UL), Crowdsourced Online Dispute Resolution
18 Ridho Reinanda (UVA), Entity Associations for Search
19 Jeroen Vuurens (UT), Proximity of Terms, Texts and Semantic Vectors in Information Retrieval
20 Mohammadbashir Sedighi (TUD), Fostering Engagement in Knowledge Sharing: The Role of Perceived Benefits, Costs and Visibility
21 Jeroen Linssen (UT), Meta Matters in Interactive Storytelling and Serious Gaming (A Play on Worlds)
22 Sara Magliacane (VU), Logics for causal inference under uncertainty
23 David Graus (UVA), Entities of Interest – Discovery in Digital Traces
24 Chang Wang (TUD), Use of Affordances for Efficient Robot Learning
25 Veruska Zamborini (VU), Knowledge Representation for Clinical Guidelines, with applications to Multimorbidity Analysis and Literature Search
26 Merel Jung (UT), Socially intelligent robots that understand and respond to human touch
27 Michiel Joosse (UT), Investigating Positioning and Gaze Behaviors of Social Robots: People's Preferences, Perceptions and Behaviors
28 John Klein (VU), Architecture Practices for Complex Contexts
29 Adel Alhuraiabi (UVT), From IT-Business Strategic Alignment to Performance: A Moderated Mediation Model of Social Innovation, and Enterprise Governance of IT
30 Wilma Latuny (UVT), The Power of Facial Expressions
31 Ben Ruijl (UL), Advances in computational methods for QFT calculations
32 Thaeer Samar (RUN), Access to and Retrieval of Content in Web Archives
33 Brigit van Loggen (OU), Towards a Design Rationale for Software Documentation: A Model of Computer-Mediated Activity
34 Maren Scheffel (OU), The Evaluation Framework for Learning Analytics
35 Martine de Vos (VU), Interpreting natural science spreadsheets
36 Yuanhao Guo (UL), Shape Analysis for Phenotype Characterisation from High-throughput Imaging
37 Alejandro Montes Garcia (TUE), WiBaf: A Within Browser Adaptation Framework that Enables Control over Privacy
38 Alex Kayal (TUD), Normative Social Applications
39 Sara Ahmadi (RUN), Exploiting properties of the human auditory system and compressive sensing methods to increase noise robustness in ASR
40 Altaf Hussain Abro (VUA), Steer your Mind: Computational Exploration of Human Control in Relation to Emotions, Desires and Social Support For applications in human-aware support systems
41 Adnan Manzoor (VUA), Minding a Healthy Lifestyle: An Exploration of Mental Processes and a Smart Environment to Provide Support for a Healthy Lifestyle
42 Elena Sokolova (RUN), Causal discovery from mixed and missing data with applications on ADHD datasets
43 Maaike de Boer (RUN), Semantic Mapping in Video Retrieval
44 Garm Lucassen (UU), Understanding User Stories - Computational Linguistics in Agile Requirements Engineering
45 Bas Testerink (UU), Decentralized Runtime Norm Enforcement
46 Jan Schneider (OU), Sensor-based Learning Support
47 Jie Yang (TUD), Crowd Knowledge Creation Acceleration
48 Angel Suarez (OU), Collaborative inquiry-based learning
-
- 2018 01 Han van der Aa (VUA), Comparing and Aligning Process Representations
02 Felix Mannhardt (TUE), Multi-perspective Process Mining
03 Steven Bosems (UT), Causal Models For Well-Being: Knowledge Modeling, Model-Driven Development of Context-Aware Applications, and Behavior Prediction
04 Jordan Janeiro (TUD), Flexible Coordination Support for Diagnosis Teams in Data-Centric Engineering Tasks
05 Hugo Huurdeman (UVA), Supporting the Complex Dynamics of the Information Seeking Process
06 Dan Ionita (UT), Model-Driven Information Security Risk Assessment of Socio-Technical Systems
07 Jieting Luo (UU), A formal account of opportunism in multi-agent systems
08 Rick Smetsers (RUN), Advances in Model Learning for Software Systems
09 Xu Xie (TUD), Data Assimilation in Discrete Event Simulations
10 Julenka Mollee (VUA), Moving forward: supporting physical activity behavior change through intelligent technology
11 Mahdi Sargolzaei (UVA), Enabling Framework for Service-oriented Collaborative Networks
12 Xixi Lu (TUE), Using behavioral context in process mining
13 Seyed Amin Tabatabaei (VUA), Computing a Sustainable Future
14 Bart Joosten (UVT), Detecting Social Signals with Spatiotemporal Gabor Filters
15 Naser Davarzani (UM), Biomarker discovery in heart failure
16 Jaebok Kim (UT), Automatic recognition of engagement and emotion in a group of children
17 Jianpeng Zhang (TUE), On Graph Sample Clustering
18 Henriette Nakad (UL), De Notaris en Private Rechtspraak
19 Minh Duc Pham (VUA), Emergent relational schemas for RDF
20 Manxia Liu (RUN), Time and Bayesian Networks
21 Aad Slootmaker (OUN), EMERGO: a generic platform for authoring and playing scenario-based serious games
22 Eric Fernandes de Mello Araujo (VUA), Contagious: Modeling the Spread of Behaviours, Perceptions and Emotions in Social Networks
23 Kim Schouten (EUR), Semantics-driven Aspect-Based Sentiment Analysis
24 Jered Vroon (UT), Responsive Social Positioning Behaviour for Semi-Autonomous Telepresence Robots
25 Riste Gligorov (VUA), Serious Games in Audio-Visual Collections
26 Roelof Anne Jelle de Vries (UT), Theory-Based and Tailor-Made: Motivational Messages for Behavior Change Technology
27 Maikel Leemans (TUE), Hierarchical Process Mining for Scalable Software Analysis
28 Christian Willems (UT), Social Touch Technologies: How they feel and how they make you feel
29 Yu Gu (UVT), Emotion Recognition from Mandarin Speech
30 Wouter Beek, The "K" in "semantic web" stands for "knowledge": scaling semantics to the web
-
- 2019 01 Rob van Eijk (UL), Comparing and Aligning Process Representations
02 Emmanuelle Beauxis Aussalet (CWI, UU), Statistics and Visualizations for Assessing Class Size Uncertainty

- 03 Eduardo Gonzalez Lopez de Murillas (TUE), Process Mining on Databases: Extracting Event Data from Real Life Data Sources
 - 04 Ridho Rahmadi (RUN), Finding stable causal structures from clinical data
 - 05 Sebastiaan van Zelst (TUE), Process Mining with Streaming Data
 - 06 Chris Dijkshoorn (VU), Nichesourcing for Improving Access to Linked Cultural Heritage Datasets
 - 07 Soude Fazeli (TUD),
 - 08 Frits de Nijs (TUD), Resource-constrained Multi-agent Markov Decision Processes
 - 09 Fahimeh Alizadeh Moghaddam (UVA), Self-adaptation for energy efficiency in software systems
 - 10 Qing Chuan Ye (EUR), Multi-objective Optimization Methods for Allocation and Prediction
 - 11 Yue Zhao (TUD), Learning Analytics Technology to Understand Learner Behavioral Engagement in MOOCs
 - 12 Jacqueline Heijerman (VU), Better Together
 - 13 Guanliang Chen (TUD), MOOC Analytics: Learner Modeling and Content Generation
 - 14 Daniel Davis (TUD), Large-Scale Learning Analytics: Modeling Learner Behavior & Improving Learning Outcomes in Massive Open Online Courses
 - 15 Erwin Walraven (TUD), Planning under Uncertainty in Constrained and Partially Observable Environments
 - 16 Guangming Li (TUE), Process Mining based on Object-Centric Behavioral Constraint (OCBC) Models
 - 17 Ali Hurriyyetoglu (RUN), Extracting actionable information from microtexts
 - 18 Gerard Wagenaar (UU), Artefacts in Agile Team Communication
 - 19 Vincent Koeman (TUD), Tools for Developing Cognitive Agents
 - 20 Chide Groenouwe (UU), Fostering Technically Augmented Human Collective Intelligence
-



Read online



https://bit.do/col_int_cg