Computational Linguistics in the Netherlands 2004

Published by
LOT
Trans 10
3512 JK Utrecht
The Netherlands

phone: +31 30 253 6006
fax: + 31 30 253 6000
e-mail: lot@let.uu.nl
http://wwwlot.let.uu.nl/

ISBN 90-76864-91-8
NUR 632

Computational Linguistics in the Netherlands 2004

Selected papers from the fifteenth CLIN meeting

Edited by
Ton van der Wouden
Michaela Poß
Hilke Reckman
Crit Cremers

LOT
Utrecht
2005

Contents

Preface

It is our pleasure to present this selection of the papers presented at the fifteenth installment of *Computational Linguistics in the Netherlands*, held at Leiden University on Friday, December 17th, 2004. Organized by the computational linguists of what was at that time called the Leiden Centre for Linguistics (ULCL), this one day conference attracted 74 abstracts from no less than four continents, and over one hundred participants.

Shalom Lappin, of King's College London, and Luc Steels, of the Free University, Brussels and the Sony Computer Science Laboratories, Paris, delivered the keynote lectures, on *From Universal Grammar to Machine Learning: the Significance of Natural Language Engineering for Theories of Cognition* and *Emergent Fluid Construction Grammars*, respectively. A written version of Prof. Lappin's paper is included in the present volume.

The separate call for papers resulted in the submission of 26 papers, of which only 16 could be included. We wish to express our heartfelt thanks to the reviewing panel consisting of Harald Baaijen, Stefano Bocconi, Antal van den Bosch, Harry Bunt, Alessandra Corda, Arthur Dirksen, Francisca de Jong, Truus Kruijt, Frank Landsbergen, Anton Nijholt, Gertjan van Noord, Jan Odijk, Manfred Sailer, Rob van der Sandt, Mariët Theune, Mark de Vos, Theo Vosse, Henk Zeevat, and Joost Zwarts, for helping us in the difficult task of selecting the papers most suitable for this volume, as well as to the authors complying with our harsh demands with respect to formatting issues and time schedules.

As you will see from the table of contents, the papers cover a wide variety of topics from the field of computational linguistics. In our view, the alphabet was the ordering principle that was the least arbitrary.

Finally, we wish to thank everyone who helped with the local organization, the former ULCL for cooperation and financial support, LOT for their willingness to publish this volume in their Occasional Series, and to the Faculty of Arts of Leiden University for providing a venue.

Leiden, October 2005

Crit Cremers
Michaela Poß
Hilke Reckman
Ton van der Wouden

Machine Learning and the Cognitive Basis of Natural Language

Shalom Lappin

Department of Philosophy, King's College London

Abstract

Machine learning and statistical methods have yielded impressive results in a wide variety of natural language processing tasks. These advances have generally been regarded as engineering achievements. In fact it is possible to argue that the success of machine learning methods is significant for our understanding of the cognitive basis of language acquisition and processing. Recent work in unsupervised grammar induction is particularly relevant to this issue. It suggests that knowledge of language can be achieved through general learning procedures, and that a richly articulated language faculty is not required to explain its acquisition.

1 Introduction

The past fifteen years have seen a massive expansion in the application of information theoretic and machine learning (ML) methods to natural language processing. This work has yielded impressive results in accuracy and coverage for engineering systems addressing a wide variety of tasks in areas like speech recognition, morphological analysis, parsing, semantic interpretation, and dialogue management. It is worth considering whether the inductive learning mechanisms that these methods employ have consequences not simply for natural language engineering, but also for our understanding of the cognitive basis of human language acquisition and processing.

A view common among computational linguists is that the information theoretic methods used to construct robust NLP systems are engineering tools. On this approach language engineering does not bear directly on the cognitive properties of grammar and human language processing. Shieber (2004 pc) suggests that the success of information theoretic methods in NLP may have implications for the scientific understanding of natural language. Pereira (2000) argues that current work on grammar induction has revived Harris' (1951), (1991) program for combining formal grammar and information theory in the study of language.

Most machine learning has used supervised learning techniques. These have limited implications for theories of human language learning, given that they require annotation of the training data with the structures and rules that are to be learned. However, recently there has been an increasing amount of promising research on unsupervised machine learning of linguistic knowledge. The results of this research suggest the computational viability of the view that general cognitive learning and projection mechanisms rather than a richly articulated language faculty may be sufficient to support language acquisition and interpretation.

In Section 2 I briefly consider the poverty of stimulus argument that has been traditionally invoked to motivate a distinct language faculty. Section 3 reviews major developments in the application of supervised machine learning to different areas of NLP. Section 4 considers some recent work in unsupervised learning of NLP systems. In Section 5 I attempt to clarify the central issues in the debate between the distinct

language faculty and generalized learning model approaches. Finally Section 6 states the main conclusions of this discussion and suggests directions for future work.

2 The Poverty of Stimulus Argument for a Language Faculty

Chomsky (1957), (1965), (1981), (1986), (1995), (2000) argues for a richly articulated language acquisition device to account for the speed and efficiency with which humans acquire natural language on the basis of "sparse" evidence. This device defines the initial state of the language learner. Its structures and constraints represent a Universal Grammar (UG) that the learner brings to the language acquisition problem. In earlier versions of the language faculty hypothesis (as in Chomsky (1965), for example) this device is presented as a schema that defines the set of possible grammars and an evaluation metric that ranks grammars conforming to this schema for a particular natural language. Since Chomsky (1981) UG has been described as an intricate set of principles containing parameters at significant points in their specification. On the Principles and Parameters (P&P) approach exposure to a very limited amount of linguistic data allows the learner to determine parameter values in order to produce a grammar for his/her language. Chomsky (2000) (p. 8) describes this Principles and Parameters model in the following terms.

> We can think of the initial state of the faculty of language as a fixed network connected to a switch box; the network is constituted of the principles of language, while the switches are options to be determined by experience. When switches are set one way, we have Swahili; when they are set another way, we have Japanese. Each possible human language is identified as a particular setting of the switches—a setting of parameters, in technical terminology. If the research program succeeds, we should be able literally to deduce Swahili from one choice of settings, Japanese from another, and so on through the languages that humans acquire. The empirical conditions of language acquisition require that the switches can be set on the basis of the very limited properties of information that is available to the child.

The language faculty view relies primarily on the poverty of stimulus argument to motivate the claim that a powerful task-specific mechanism is required for language acquisition. According this argument the complex grammar that a child achieves within a short period, with very limited data cannot be explained through general learning procedures of the kind involved in other cognitive tasks.

This is, at root, a "What else could it be?" argument. It asserts that, given the complexity of grammatical knowledge and the lack of evidence for discerning its properties, we are forced to the conclusion that much of this knowledge is not learned at all but must already be present in the initial design of the language learner. The basis for this claim is the assumption that first language learners have access to very limited amounts of data, where this data is, in general, free of negative information (corrections), and inadequate to support inductive projection of grammars under the attested conditions of acquisition. In fact, this assumption has been increasingly subject to

effective challenges. So, for example, Pullum and Scholz (2002) argue that there is no poverty of stimulus in language learning. The linguistic data to which children are exposed is far richer than poverty of stimulus theorists suggest. Similarly Chouinard and Clark (2003) present the results of detailed case studies showing that parents provide children with a wealth of negative evidence that plays a significant role in the language acquisition process.

One of Chomsky's (1957) original arguments against statistical induction of grammar turns on the absence of many (most) grammatical structures in corpora. This is an instance of the sparse data problem. Pereira (2000) points out that smoothing techniques, introduced in Good (1953), permit the assignment of probability values to unobserved linguistic events. When enriched with smoothing, statistical modelling of NL learning can deal effectively with sparse data. This development undermines the poverty of stimulus argument from the perspective of the power of task-general computational devices for inductive learning. A plausible alternative to the UG hypothesis is that, given reasonable initial settings for a set of linguistic categories and a search space for grammar rules, general learning strategies of the sort used in AI and NLP can account for language acquisition and processing without the assumption of a task-specific language learning device.

3 Supervised Learning

In supervised learning a corpus is annotated with the structures or features that the system must learn to recognize. This corpus provides the gold standard for training and evaluation. Symbolic machine learning algorithms extract rules or classifying procedures from the corpus to identify the marked properties in unlabelled corpora. Statistically driven learning methods construct models from the training corpus that determine the probability distributions for the marked properties over a set of possible linguistic contexts.

Supervised acquisition of part of speech (POS) tags has produced successful broad coverage taggers. Church (1988), (1992) proposes a stochastic POS tagger. Brill (1992), (1994) constructs a transformation-based tagger that applies rule-based machine learning. These systems and other current taggers generally have a lower bound of 96% accuracy evaluated against POS annotated corpora like the British National Corpus (BNC) and the Penn Tree Bank.

Probabilistic parsing is another domain in which supervised learning has yielded impressive results Charniak (1997) and Collins (1998) develop Probabilistic Context Free Grammars (PCFGs) which extract CFG rules with specified probability values from the Penn Tree Bank. The PCFG rules are lexicalized to identify the lexical head of a constituent and dependency relations. Charniak's PCFG includes lexical subcategorization features. Collins' grammar represents argument-adjunct distinctions. Both grammars achieve recall and precision scores of close to 90% on the Wall Street Journal (WSJ) of the Penn Tree Bank.

Clark and Curran (2004) describe a Maximum Entropy statistical Combinatory Categorical Grammar (CCG) parser. The CCG parser represents unbounded dependencies as well as local function-argument structures. It is trained on a corpus an-

notated with CCG lexical tags and lexical dependency structures (Hockenmaier and Steedman (2002)). Dependency structures are represented as features, and the system computes a model of the most likely set of dependencies, given a sentence, from the annotated gold standard. The authors report recall and precision scores of over 86% for labelled dependencies and over 92% for unlabelled dependencies on a test set from the WSJ.

Similar methods have been used for wide coverage semantic representation. Bos, Clark, Curran, Hockenmaier and Steedman (2004) construct a system for assigning logical forms to the parse structures of a statistical CCG. Typed λ-terms are assigned to tagged lexical items, and λ-terms are composed for phrases in accordance with the dependency relations of the parse structure. The reduced λ-terms for sentences are first-order formulas with Davidsonian event variables. The authors report that the system assigns well-formed logical forms to 92% of parse input when tested on WSJ text. These logical forms are not sufficiently expressive for many sentences. They do not handle higher-order quantificational determiners or intensional modifiers. However, the system could be extended to generate more realistic representations to deal with these phenomena.

Supervised machine learning is also being fruitfully applied to ellipsis resolution. Nielsen (2003) uses a set of alternative machine learning algorithms to identify elided VP's in a sample of the BNC on the basis of contexts specified with POS features. Transformation Based Learning (TBL) obtained an F score (a weighted average of recall and precision) of 76.61%, and a Maximum Entropy system yielded 76.01%. Nielsen (2004) tests a subset of these algorithms on a much larger corpus containing text from both the BNC and the Penn Tree Bank, and using full parse structures as input. The Maximum Entropy system achieved an F score of approximately 71% for this text.

Liakata and Pulman (2004) propose a method for learning a domain theory from a text by Inductive Logic Programming (ILP). The theory consists of a set of Horn clause rules that express relations among the main properties, relations, and entities cited in the text. It is encoded as a probabilistic Finite State Automata whose transition arcs are labelled with simple Horn clauses that are assigned probability values. The method is applied to text from the Penn Tree Bank and from the ATIS corpus.

Fernández, Ginzburg and Lappin (2005) apply supervised machine learning techniques to the task of identifying the interpretational type of non-sentential utterances (NSUs) in dialogue from a set of possible readings. They construct a procedure for automatically annotating a corpus of 1109 NSUs, extracted from the BNC, with 9 features. The automatic feature annotation achieves 89% accuracy when evaluated against a randomly selected 10% sample of the NSU corpus. They apply four machine learning algorithms, C4.5-based decision trees (Quinlan (1993)), SLIPPER, a greedy rule learning system with confidence rated rule boosting (Cohen and Singer (1999)), TiMBL, a memory-based learner (Daelemans, Zavrel, van der Sloot and van den Bosch (2003)), and a maximum entropy system (Le (2003)), to this annotated data set. The four ML algorithms achieve F scores of between 87% and 90%.

The striking achievements of supervised learning in NLP show that powerful symbolic and statistical induction procedures applied to corpora can acquire knowledge of

the structure and interpretation of NL quickly and efficiently. However, these procedures require that the information to be learned is explicitly represented in the training data. This information defines the initial conditions from which learning proceeds. Therefore the success of supervised learning in NLP does not, in itself, provide decisive evidence against the assumption of a language specific learning device.

4 Unsupervised Learning

In unsupervised learning the training corpus is not annotated with the structures or features to be acquired. The search space is constrained to a set of possible entities to be learned (such as lexical classes, constituent structures, CFG rules, etc.). Learning is achieved largely through the identification of distributional and clustering patterns by which classes of similar objects are recognized. Successful acquisition of linguistic structure and content through unsupervised methods would provide significant motivation for the view that weak assumptions concerning linguistic structure or rule hypothesis space, combined with general cognitive mechanisms of induction and projection are sufficient for language learning.

Goldsmith (2001) uses Minimal Description Length (MDL) criteria to select between alternative morphological analyses of words in unmarked text. He employs several probabilistic methods as heuristic procedures to generate morphological signatures that split the words of a corpus into stems and suffixes. Goldsmith uses MDL to identify the optimal morphological system as the one which provides the most compressed description of the full range of data with the most compact set of signatures. He reports that in a test set of 1000 alphabetically consecutive words taken from a 500,000 word English corpus 82.9% of the analyses that his system produces are good.

Schone and Jurafsky (2001) improve on Goldsmith's results with an algorithm for unsupervised morphological analysis of stems and affixes that combines three main factors to identify pairs of morphologically related words. It makes use of induced semantic relations among lexical items, orthographic connections among words measured in terms of transformability operations, and local syntactic contexts of distribution common to attested morphological variants. It also uses weighted transitive closures of identified morphological relations among words to extend sets of variants. Schone and Jurafsky test their algorithm on English (6.7 million words of newswire text), German (2.3 million words), and Dutch (6.7 million words). They use the hand annotated CELEX lexicon for each language as the gold standard for evaluation. Their algorithm gives an F score of 88.1 % for English suffixes, compared to 81.8% that Goldsmith's system achieves for the same text, 92.3% for German (Goldsmith 84%), and 85.8% for Dutch (Goldsmith 75.8%).

Clark (2000) describes an unsupervised distributional method for identifying lexical syntactic categories through clustering. The tag set which this method generates compares favourably to the POS tag set of the BNC (CLAWS). Clark reports that a probabilistic finite state model for tagging gave lower perplexity values for the tag set obtained by unsupervised learning than for CLAWS, showing that the unsupervised tags express significant distributional generalizations. Reliable unsupervised POS tagging provides the basis for unsupervised grammar acquisition.

Clark (2001) describes an unsupervised system that learns a stochastic CFG from a tagged text using distributional information concerning local contexts for tag sequences. This system gives a somewhat disappointing F score of 41% on the ATIS corpus.

The grammar induction system proposed by Klein and Manning (2002) is an unsupervised method that learns constituent structure from part of speech (POS) tagged input by assigning probability values to sequences of tagged elements as constituents in a tree. They bias their model to parse all sentences with binary branching trees, and they use an Expectation Maximization (EM) algorithm to identify the most likely tree structure for a sentence. Their method relies on recognizing (unlabelled) constituents through distributional clustering of corresponding sequences in the same contexts, where a tree structure is constrained by the requirement that sister constituents do not overlap (have non-null intersections of elements).

The Klein and Manning procedure achieves an F score of 71% on WSJ text, using Penn Tree Bank parses as the standard of evaluation. This score is impressive when one considers a limitation that the evaluation procedure imposes on their system. The upper bound on a possible F-score for their algorithm is 87% because the Penn treebank assigns non-binary branching to many constituents. In fact, many of the system's "errors" are linguistically viable parses that do not conform to analyses of the Penn Treebank. So, for example, the Treebank assigns flat structure to NPs, while the Klein and Manning procedure analyzes NPs as having iterated binary branching. Parses of the latter kind can be motivated on linguistic grounds.

One might object to the claim that Klein and Manning's parser is genuinely unsupervised on the grounds that it uses the POS tagging of the Penn Treebank as input. They run an experiment in which they apply their procedure to WSJ text annotated by an unsupervised tagger, and obtain an F score of 63.2%. However, as they point out, this tagger is not particularly reliable. Other unsupervised taggers, like the one that Clark (2000) describes, yield very encouraging results, and outputs of these taggers might well permit the parser to perform at a level comparable to that which it achieves with the Penn Treebank tags.

Klein and Manning (2004) describe a probabilistic model for unsupervised learning of lexicalized head dependency grammars. The system assigns probabilities to dependency structures for sentences by estimating the likelihood that each word in the sentence is a head that takes a specified sequence of words to its left and to its right as argument or adjunct dependents. The probabilities are computed on the basis of the context in which the head appears, where this context consists of the words (word classes) occurring immediately on either side of it. Like the constituent structure model, their dependency structure model imposes binary branching as a condition on trees. The procedure achieves an F score of 52.1% on Penn Treebank test data. This result underrates the success of the dependency model to the extent that it relies on strict evaluation of the parser's output against the dependency structures of the Penn Treebank, in which NPs are headed by N's. Klein and Manning report that in many cases their dependency parser identifies the determiner as the head of the NP, and this analysis is, in fact, linguistically viable.

When the dependency system is combined with their unsupervised constituency

grammar, the integrated model outperforms each of these systems. In the composite model the score for each tree is computed as the product of the individual models that the dependency grammar and the constituency structure grammar generate. This model uses both constituent clustering and the probability of head dependency relations to predict binary constituent parse structure. It yields an F score of 77.6% with Penn Treebank POS tagging. It also achieves an F score of 72.9% with an unsupervised tagger (Schuetze (1995)).

This work on unsupervised grammar induction indicates that it is possible to learn a grammar that identifies complex syntactic structure with a relatively high degree of accuracy using a model containing a weak assumptions concerning syntactic structure, specifically the restriction of binary branching, a non-overlap constraint for constituents, and limited conditions on head argument/adjunct dependency relations.

Recent research on unsupervised grammar induction offers support for the view that knowledge of language can be achieved through general machine learning methods on the basis of a minimal set of initial settings for possible linguistic categories and rule hypotheses. This work suggests a sequenced boot strap model of language learning in which each level of structure acquired provides the input to a higher successor component of grammar. In at least some cases both the basic categories and the hypothesis space might be derived from more general cognitive processing patterns (like the binary branching trees that Klein and Manning (2002), (2004) generate for constituent and dependency structures).

By contrast to machine learning NLP only a few small scale prototypes for grammar acquisition with the P&P model have been implemented, most notably by Fong (1991), (2005). They have not been extensively applied to real data from linguistic corpora. No robust, wide coverage systems using this model have been designed or tested. A common response of P&P advocates to this criticism is to argue that they are concerned with characterizing the set of possible languages rather than to develop broad coverage parsers for particular languages. This is, in effect, a circular argument, as it reduces to the assertion that the proper objective of linguistic theory is to specify the properties of the language faculty that constitute UG. But it is the existence of such a faculty, and hence the motivation for this enterprise that is at issue in the current discussion.

Some critics of the view that machine learning can provide a viable model of human grammar acquisition have argued that an ML system learns only from a corpus of grammatical sentences, and so it is limited to recognizing well formed phrases.[1] This claim is misconceived. The parser that an ML system produces can be engineered as a classifier to distinguish grammatical and ungrammatical strings, and it can identify the structures assigned to a string under which this distinction holds. Such a classifier can also be refined to provide error messages that identify those properties of an unsuccessful parse that cause a sentence to be identified as ungrammatical. It is important to note that such a classifier is generated by machine learning applied to a data set, where the learning task is defined by the (relatively) weak grammatical assumptions

[1] See, for example, Carson Schutze's contributions of April 20 and May 5, 2005, on the *Linguist List*, to the discussion of Richard Sproat and Shalom Lappin, "A Challenge to the Minimalist Community", *Linguist List*, April 5, 2005.

of the language model that the learner invokes.

5 Clarifying the Issues of the Discussion

It has occasionally been suggested that the debate between ML-based NLP and P&P involves a choice between symbolic and non-symbolic approaches to computing. In fact the role of symbolic computing methods in NLP is not an issue in this discussion. Many machine learning algorithms produce rule systems for classification, as is the case with TBL, ILP, and SLIPPER. Statistically driven learning procedures apply to data annotated with high-level linguistic information and, in many cases, produce abstract representations, including syntactic structures, semantic role information, and logical forms.

It is important to recognize that the need to assume a rich set of innate learning principles is not in dispute. The machine learning approach posits a powerful induction and projection device with initial categories and hypothesis settings. However, unlike the language faculty view, it takes most of this innate mechanism to be a set of general cognitive capacities that apply to a wide variety of learning and processing problems rather than a task-specific device that applies only to natural language learning. The constraints that the model imposes on the set of grammatical structures are minimal and define a large space of possible languages and grammars.

The debate does not concern "mentalism". The role of internal states and operations of the language learner is not in question. The machine learning approach requires them as part of its model of the computations involved in learning and processing. Whether these states and operations are conceived of as mental properties and events or as neurological phenomena is not relevant to either side of the debate.

The primary question at issue between the ML and P&P approaches is the status of the poverty of stimulus argument as the basis for postulating a highly articulated language faculty (UG) for language learning. The success of machine learning methods shows that this argument does not go through. The fact that no robust P&P system for acquiring grammar from real data has yet been implemented casts serious doubt on the computational credibility of this approach to language acquisition. Refined information theoretic methods provide viable computational procedures for acquiring linguistic knowledge.

However, it is important to recognize that the success of unsupervised ML grammar induction does not, in itself, entail that these procedures actually apply in human language acquisition. To understand human language learning and interpretation it is necessary to achieve deeper insight into the psychological and neurological facts of the acquisition process.

6 Conclusions

We have seen that advances in the development of information theoretic learning methods have given rise to substantial progress in the computational modelling of a wide range of NLP tasks. This work is commonly regarded as an achievement in natural language engineering. In fact, it also has implications for theories of human

language acquisition and processing. Recent work in unsupervised grammar acquisition is particularly significant in indicating the possibility of accounting for language learning through general procedures of induction and probabilistic learning.

Much remains to be done in order to further clarify the issues that have been raised here. Advocates of the general inductive learning approach must focus on the refinement and further development of unsupervised learning procedures for NLP tasks. Proponents of the language-specific learning device view should be concerned to implement precise and convincing versions of a P&P model for particular grammar acquisition tasks. To be credible these systems must achieve the same level of robustness over large corpora that ML acquisition and processing systems have succeeded in demonstrating. In order to determine the cognitive reality of their respective models adherents of both approaches must work more closely with psychologists and neuroscientists to explore the relation of computationally viable models of language learning and processing to human performance.

7 Acknowledgements

Earlier versions of this paper were presented at the University College London Workshop on Computational Linguistics, November 2004; Computational Linguistics in the Netherlands, Leiden, December, 2004; the Linguistics Colloquium of the University of Illinois at Urbana-Champaign, March, 2005; and the Construction of Meaning Workshop, Stanford, April, 2005. I am grateful to the participants of these forums for useful comments and criticism. I would also like to thank Eve Clark, Jennifer Cole, Crit Cremers, Jonathan Ginzburg, Dan Jurafsky, Chris Manning, John Nerbonne, Leif Nielsen, Dan Roth, Ivan Sag, and James Yoon for helpful discussion of some of the ideas presented here. I am particularly grateful to Stuart Shieber and Richard Sproat for conversations that were invaluable to me in clarifying my thinking on many of the issues addressed in this paper. Of course I bear sole responsibility for its content and any errors it may contain.

References

Bos, J., Clark, S., Curran, J., Hockenmaier, J. and Steedman, M.(2004), Wide-coverage semantic representations from a CCG parser, *Proceedings of COLING 18*, Geneva, Switzerland.

Brill, E.(1992), A simple rule-based part of speech tagger, *Proceedings of the Third Conference on Applied Natural Language Processing*, Trento, Italy, pp. 152 –155.

Brill, E.(1994), Some advances in transformation-based part of speech tagging, *Proceedings of the Twelfth National Conference on Artificial Intelligence*, pp. 722–727.

Charniak, E.(1997), Statistical parsing with context-free grammar and word statistics, *Proceedings of the Fourteenth National Conference on Artificial Intelligence*, pp. 598–603.

Chomsky, N.(1957), *Syntactic Structures*, Mouton, The Hague.

Chomsky, N.(1965), *Aspects of the Theory of Syntax*, MIT Press, Cambridge, MA.
Chomsky, N.(1981), *Lectures on Government and Binding*, Foris, Dordrecht.
Chomsky, N.(1986), *Knowledge of Language: Its Nature, Origin, and Use*, Praeger, New York.
Chomsky, N.(1995), *The Minimalist Program*, MIT Press, Cambridge, MA.
Chomsky, N.(2000), *New Horizons in the Study of Language and Mind*, Cambridge University Press, Cambridge.
Chouinard, M. and Clark, E.(2003), Adult reformulations of child errors as negative evidence, *Journal of Child Language* **30**, 637–669.
Church, K.(1988), A stochastic parts program and noun phrase parser for unrestricted text, *Second Conference on Applied Natural Language Processing*, Austin, TX, pp. 136–143.
Church, K.(1992), Current practice in part of speech tagging and suggestions for the future, *in* Simmons (ed.), *Sbornik Praci: In Honor of Henry Kucera*, Michigan Slavic Studies, Michigan, pp. 13–48.
Clark, A.(2000), Inducing syntactic categories by context distribution clustering, *Proceedings of CoNLL 2000*, Lisbon, Portugal.
Clark, A.(2001), Unsupervised induction of stochastic context-free grammars using distributional clustering, *Proceedings of CoNLL 2001*, Toulouse, France.
Clark, S. and Curran, J.(2004), Parsing the WSJ using CCG and log-linear models, *Proceedings of the Forty-Second Meeting of the Association for Computational Linguistics*, Barcelona, Spain, pp. 104–111.
Cohen, W. and Singer, Y.(1999), A simple, fast, and effective rule learner, *Proceedings of the 16th National Conference on Artificial Intelligence (AAAI-99)*.
Collins, M.(1998), *Head-Driven Statistical Models for Natural Language Parsing*, PhD thesis, University of Pennsylvania.
Daelemans, W., Zavrel, J., van der Sloot, K. and van den Bosch, A.(2003), TiMBL: Tilburg Memory Based Learner, v. 5.0, Reference Guide, *Technical Report ILK-0310*, University of Tilburg.
Fernández, R., Ginzburg, J. and Lappin, S.(2005), Using machine learning for non-sentential utterance classification, *Proceedings of the Sixth SIGdial Workshop on Discourse and Dialogue*, Lisbon, pp. 77–86.
Fong, S.(1991), *Computational Properties of Principled-Based Grammatical Theories*, PhD thesis, Massachuetts Institute of Technology.
Fong, S.(2005), Computation with probes and goals: A parsing perspective, *in* A. D. Sciullo and R. Delmonte (eds), *UG and External Systems*, John Benjamins, Amsterdam.
Goldsmith, J.(2001), Unsupervised learning of the morphology of a natural language, *Computational Linguistics* **27**, 153–198.
Good, I.(1953), The population frequencies of species and the estimation of population parameters, *Biometrika* **40**, 237–264.
Harris, Z.(1951), *Structural Linguistics*, University of Chicago Press, Chicago, IL.
Harris, Z.(1991), *A Theory of Language and Information: A Mathematical Approach*, Clarendon Press, Oxford, New York.
Hockenmaier, J. and Steedman, M.(2002), Generative models for statistical parsing

with combinatory categorial grammar, *Proceedings of the Fortieth Meeting of the Association for Computational Linguistics*, Philadelphia, PA, pp. 335–342.

Klein, D. and Manning, C.(2002), A generative constituent-context model for improved grammar induction, *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pp. 128–135.

Klein, D. and Manning, C.(2004), Corpus-based induction of syntactic structure: Models of dependency and constituency, *Proceedings of the 42th Annual Meeting of the Association for Computational Linguistics*, Barcelnoa, Spain.

Le, Z.(2003), Maximum Entropy Modeling Toolkit for Python and C++. http://homepages.inf.ed.ac.uk/ s0450736/maxent_toolkit.php.

Liakata, M. and Pulman, S.(2004), Learning theories from text, *Proceedings of COLING 18*, Geneva, Switzerland.

Nielsen, L.(2003), Using machine learning techniques for VPE detection, *Proceedings of Recent Advances in Natural Lanuage Processing*, Borovets, Bulgaria, pp. 339–346.

Nielsen, L.(2004), Verb phrase ellipsis detection using automatically parsed text, *Proceedings of COLING 18*, Geneva, Switzerland.

Pereira, F.(2000), Formal grammar and information theory: Together again?, *Philosophical Transactions of the Royal Society*, Royal Society, London, pp. 1239–1253.

Pullum, G. and Scholz, B.(2002), Empirical assessment of stimulus poverty arguments, *The Linguistic Review* **19**, 9–50.

Quinlan, R.(1993), *C4.5: Programs for Machine Learning*, Morgan Kaufmann, San Francisco, CA.

Schone, P. and Jurafsky, D.(2001), Knowledge-free induction of inflectional morphologies, *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL-2001)*, Pittsburgh, PA.

Schuetze, H.(1995), Distributional part-of-speech tagging, *in* 141-148 (ed.), *Proceedings of the European Chapter of the Association for Computational Linguistics (EACL 7)*.

When the Hothead Speaks

Simulated Annealing Optimality Theory for Dutch Fast Speech

Tamás Bíró

Humanities Computing, University of Groningen

Abstract

Simulated Annealing, a wide-spread technique for combinatorial optimisation, is employed to find the optimal candidate in a candidate set, as defined in *Optimality Theory* (OT). Being a heuristic techniques, simulated annealing does not guarantee to return the correct solution, and yet, some result is always returned within a constant time. Similarly to language production, this time framework can be diminished with the cost of diminishing correctness. We demonstrate how simulated annealing can model linguistic performance, built upon a competence theory, namely, OT. After having applied simulated annealing to OT, we attempt to reproduce empirical observations on metrical stress in Dutch fast speech. Simulated annealing necessitates defining a *topology* on the candidate set, as well as an exact formulation of the constraint OUTPUT-OUTPUT CORRESPONDENCE.

1 Introduction: OT and optimisation

Optimality Theory (OT; Prince and Smolensky (1003), aka Prince and Smolensky (2004)) has been an extremely popular model in linguistics in the last decade. The architecture of an OT grammar, as shown in Figure 1, is composed of two parts. Out of the input (the underlying representation *UR*), the GEN module generates a set of candidates (*GEN*(*UR*)), each of which is evaluated by the EVAL module, and the best element is returned as the output (the surface representation *SR*).

EVAL is usually seen as a pipeline, in which the *constraints* filter out the sub-harmonic candidates. Each constraint assigns violation marks to the candidates in its input, and candidates that have more marks than some other ones are out of the game. Alternatively, EVAL can also be seen as a function assigning a harmony value to the candidates, the most harmonic of which will surface in the language. This *Harmony function* has a remarkable property: being worse on a higher ranked constraint can never be compensated by a good behaviour on a lower ranked constraint. This phenomenon, referred to as *the categorical ranking of the constraints*, or as the *Strict Domination Hypothesis*, follows from the filtering approach: whoever is filtered out at an earlier stage never comes back.

The traditional way of representing the competing candidates is to use a *tableau*, such as the one in (1). The left column contains the elements of the candidate set, that is, *GEN*(UR). For a given candidate w_i, the number of violation marks $C_j(w_i)$—in most cases a non-negative integer—assigned by constraint C_j is given, and the exclamation mark brings the attention to the point where a given candidate meets its

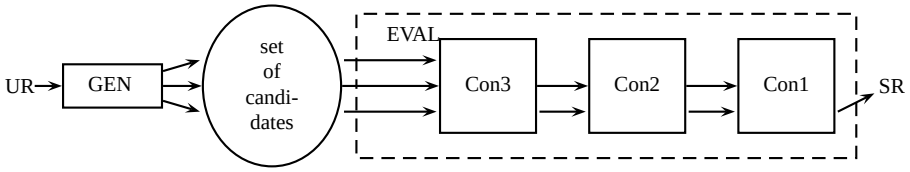


Figure 1: The basic architecture of an Optimality Theoretic grammar

Waterloo. The ☞ symbol points to the winning candidate.

(1)

/UR/	C_n	C_{n-1}	...	C_{k+1}	C_k	C_{k-1}	C_{k-2}	...
☞w_1	2	0		1	2	3	0	
w_2	2	0		1	3 !	1	2	
w_3	3 !	0		1	3	1	2	

If the constraints are functions mapping from the candidate set *GEN*(*UR*) to the set of non-negative integers ($\mathbb{N}_0$), then the EVAL module can be seen as an *Eval function* that assigns a vector—a *violation profile*, an (inverse) *Harmony value*, which is a shorthand for a row in a tableau—to each of the candidates:

(2) $$E(w) = \big(C_n(w), C_{n-1}(w), ..., C_1(w)\big) \in \mathbb{N}_0^n$$

together with an *optimisation algorithm*. The role of the optimisation algorithm is to find the optimal element of the candidate set, and to return it as the output (surface representation, SR, that is the grammatical form[1]):

(3) $$\mathit{SR}(\mathit{UR}) = \mathit{argmin}_{w \in \mathit{Gen}(\mathit{UR})} E(w)$$

Here, optimisation is with respect to *lexicographic ordering*, for this is the ordering realising the *categorical ranking* (strict hierarchy) of the constraints. *Lexicographic ordering* of vectors is the way words are sorted in a dictionary (e.g. *abacus*, *abolish*,..., *apple*,..., *zebra*): first compare the first element of the vectors, then, if they are the same, compare the second one, and so on. Formally speaking:

$E(w_1) > E(w_2)$, if there exists $k \in \{n, n-1, ..., 1\}$ such that

1. $C_k(w_1) > C_k(w_2)$, and
2. for all $j \in \{n, n-1, ..., 1\}$, if $j > k$ then $C_j(w_1) = C_j(w_2)$.

Constraint C_k, which determines the relative ordering of $E(w_1)$ and $E(w_2)$, will be called the *fatal constraint* (the highest ranked constraint with uncancelled marks).

[1]The form appearing in the language is not always the output of the OT grammar itself, but a trivial function F of it. For instance, parsing brackets may have to be removed. However, the inverse of the function F is not always functional, thus sometimes more outputs (parses) may describe the same observed phenomenon, posing a challenge to learning algorithms (Tesar and Smolensky 2000).

Furthermore, if $E(w_1) < E(w_2)$, than we shall say that candidate w_1 is *better* (*more harmonic*) than candidate w_2 ($w_1 \succ w_2$). A more detailed mathematical analysis is presented in Bíró (2005) and in Bíró (forthcoming)

The computational challenge posed by Optimality Theory is to realise the optimisation algorithm required by EVAL. Indeed, Eisner (2000) demonstrates that finding the optimal candidate (*generation* in OT) is OptP-complete. In addition, the candidate set is infinite in numerous linguistic models. Several solutions have been proposed, although each of them is built upon certain presuppositions, and they also require large computational resources. Finite state techniques (see references in Bíró (2003)) not only require GEN and constraints to be finite state, but work only with some further restrictions. The presuppositions of *Chart parsing* (*dynamic programming*, e.g. Tesar and Smolensky (2000), Kuhn (2000)) are more likely to be met by most linguistic models, yet it also makes use of a relatively large memory.

If our goal is, however, to find an optimisation technique which is cognitively adequate, we do not need an exact algorithm. Indeed, speech contains frequently performance errors.

The optimisation algorithm should, under normal conditions, find the "correct", i.e. the grammatical output—the optimal element of the candidate set—with high probability. Even more, the output is returned in constant time, since the partner in a conversation is not a computer user watching the sandglass. Further, human speakers sometimes speed up the computational algorithm, and the price paid is precision. We propose to see (some) fast speech phenomena (performance errors) as decreased precision (erroneous outputs) of the optimisation algorithm in EVAL, due to the increased speed.

This train of thought leads us straightforward to heuristic optimisation techniques, defined by Reeves (1995) as "*a technique which seeks good (i.e. near-optimal) solutions at a reasonable computational cost without being able to guarantee either feasibility or optimality, or even in many cases to state how close to optimality a particular feasible solution is.*" In the present paper, we implement Optimality Theory by using the simplest heuristic optimisation technique, *simulated annealing*.

We will see that simulated annealing meets all our criteria. Its computational requirements are minimal, compared to most other methods, and it returns a "good (i.e. near-optimal) solution" of even an NP-complete problem in limited time. This time interval can be reduced by paying on precision. In particular, by observing changes in the stress patterns in Dutch fast speech, we demonstrate how a proper competence grammar can produce correct outputs under normal conditions, but starts making human-like errors under time pressure. Thereby, we argue for the cognitive adequateness of the *Simulated Annealing Optimality Theory Algorithm* (SA-OT).

2 Simulated Annealing: a heuristic optimisation technique

Simulated annealing, also called as *Boltzmann Machines*, is a wide-spread stochastic technique for combinatorial optimisation (Kirkpatrick, Jr. and Vecchi 1983). It performs a random walk in the search space, and differs from *gradient descent* by allowing uphill moves—thereby escaping local minima—with a probability that de-

creases during the simulation. Only few have applied it in linguistics, for instance in parsing (Howells 1988, Kempen and Vosse 1989, Selman and Hirst 1994). Simulated annealing is also found in the pre-history of Optimality Theory (Smolensky 1986).

The idea originates in solid state physics. An interstitial defect in a crystal lattice corresponds to a local minimum in the energy E of the lattice. Although the perfect lattice would minimise the energy, the defect is stable, because any local change increases E. In order to reach the global minimum, one needs either to globally restructure the lattice within one step, or to be permitted to temporarily increase the energy of the lattice.

Heating the lattice corresponds to the second option. The lattice is allowed "to borrow" some energy, that is, to transform provisionally thermic energy into the binding energy of the lattice, thereby climbing the energy barrier separating the local minimum from the global minimum. At temperature T, the probability of a change that increases the lattice's energy by ΔE is $e^{\frac{-\Delta E}{kT}}$, where $k = 1.38 \times 10^{-23} JK^{-1}$ is Boltzmann's constant. The higher the temperature, the bigger energy jumps ΔE are allowed.

Annealing a metal means heating it to a high temperature, and then cooling it down slowly. The lower the temperature, the lower energy hills the system is able to climb; thus it gets stuck in some valley. At the end of the annealing, the system arrives at the bottom of the valley reached. With a slower cooling schedule, the likelihood of finding the valley including the global minimum is higher.

Now, the idea of *simulated annealing* is straightforward (cf. eg. Reeves 1995). We search for the state of a system minimising the quantity E (Energy or Evaluation) by performing a random walk in the search space. If the rule were to move always downhill (*gradient descent*), we would quickly get stuck in local minima. This is why we also allow moving upwards ("borrowing thermic energy") with some chance, which is higher in the beginning of the simulation, and which then diminishes.

For this purpose, a fictive "temperature" T is introduced. The random walk starts from an initial state w_0. At each step of the simulation, we randomly pick one of the neighbouring states (w') of the actual state w (cf. Fig. 2). Thus, a *topology* on the search space has to define the neighbours of a state (the *neighbourhood structure*), as well as the *a priori probability distribution* determining which neighbour to pick. Subsequently, we compare w' to w, and the random walker moves to w' with *transition probability* $P(w \to w' \mid T)$, where T is the temperature at that moment of the simulation. If $E(w)$ is the function to minimise, then:

$$P(w \to w' \mid T) = \begin{cases} 1 & \text{if } E(w') \leq E(w) \\ e^{-\frac{E(w')-E(w)}{T}} & \text{if } E(w') > E(w) \end{cases} \tag{4}$$

Moving downhill is always possible, and moving uphill depends on the difference in E and on the actual temperature T. At the beginning of the simulation, T is assigned a high value, making any move very likely. The value of T is then decreased gradually, while even the smallest jump does not become highly improbable. When the temperature reaches its lowest value, the algorithm returns the state into which the random walker is "frozen" finally—this is a local minimum. Obviously, nothing guarantees

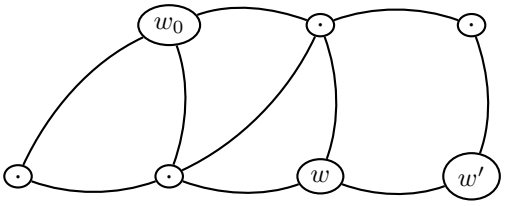


Figure 2: A schematic view of the search space—in SA-OT, the candidate set with a topology (neighbourhood structure)—in which simulated annealing realises a random walk.

finding the global minimum, but the slower the *cooling schedule* (the more iterations performed), the higher the probability of finding it.

3 Simulated Annealing for Optimality Theory

How to implement simulated annealing to Optimality Theory? The search space is the candidate set, defined by standard Optimality Theory. Yet, a *neighbourhood structure* (a *topology*) should be added to it (Fig. 2). which determines the picking of the next candidate w'. We propose to consider two candidates neighbours if they differ only minimally, that is, if a *basic operation* transforms one into the other. The algorithm gets stuck in local optima, candidates better than their neighbours. Thus, the definition of the topology influences crucially which candidates are returned besides the global optimum; these forms will be predicted to be the performance errors or the fast speech forms.

Enriching a model with further concepts—adding a topology to standard OT—could diminish the strength of a model. Yet, we have here a larger set of observations: not only the grammatical forms, but also speech errors and their frequencies. Standard OT predicts the grammatical form to be the globally optimal candidate, whereas the neighbourhood structure added to it accounts for performance errors. It is in a very non-trivial way that the interaction of the topology, the constraint hierarchy and the cooling schedule determines which local optimum is returned with what probability. Consequently, finding a simple, convincing—non *ad hoc*—topology reasonably accounting for the observed data is not a self-evident task.

If the topology determines the horizontal structure of the landscape in which the random walker roves, then the Harmony function to be optimised contributes its vertical structure. Here again, traditional Optimality Theory provides only the first part of the story. The transition probability $P(w \to w' \mid T) = 1$, if w' is better than w ($w' \succ w$, that is, $E(w') < E(w)$). But how to define the transition probability to a worse candidate, in function of the actual temperature T?

We begin by understanding the meaning of "temperature" in simulated annealing. According to equation (4), T defines the range of $E(w') - E(w)$ above which uphill moves are prohibited ($P(w \to w' \mid T) \approx 0$, if $E(w') - E(w) \gg T$), and below which they are allowed ($P(w \to w' \mid T) \approx 1$, if $E(w') - E(w) \ll T$).

In turn, our agenda is the following: first we define the difference of two violation

profiles ($E(w') - E(w)$), then define temperature in an analogous way, and last adjust the definition (4).

The difference of two violation profiles seen as vectors (cf. (2)) is simply:

$$E(w') - E(w) = (C_n(w') - C_n(w), ..., C_1(w') - C_1(w)) \tag{5}$$

Yet, what interests us when comparing two candidates is only the fatal constraint (the highest ranked constraint with uncancelled violation marks). The general structure of Optimality Theory teaches us to neglect the lower ranked constraints.[2] Therefore, we define the *magnitude* of any vector $(a_n, ..., a_1)$ as

> $\|(a_n, ..., a_1)\| = \langle k, a_k \rangle$, where k is the lowest element of $\{n, ..., 1\}$ such that $\forall j \in \{n, ..., 1\}$: if $j > k$, then $a_j = 0$.
> Moreover, $\|(0, 0, ..., 0)\| = \langle 0, 0 \rangle$.

We shall use not the difference (5), but rather the magnitude of the difference of the violation profiles, $\|E(w') - E(w)\| = \langle k, C_k(w') - C_k(w) \rangle$ in simulated annealing. Take the following tableau to exemplify this idea:

	C_n	C_{n-1}	...	C_{k+1}	C_k	C_{k-1}	C_{k-2}	...
$E(w')$	2	0		1	2	3	0	
$E(w)$	2	0		1	3	1	2	
$E(w') - E(w)$	0	0		0	-1	2	-2	

Here, $\|E(w') - E(w)\| = \langle k, -1 \rangle$, since C_k is the fatal constraint, the highest constraint with uncancelled marks. We may ignore constraints ranked below C_k.

In short, the difference (5) of two violation profiles could be reduced from an n-tuple (n-dimensional vector) to a pair $\langle k, C_k(w') - C_k(w) \rangle$. The Strict Domination Hypothesis does not allow reducing it to a single real number, however (Bíró forthcoming).

In the next step, we introduce temperature. As explained, the role of temperature in simulated annealing is to gradually decrease the transition probability to a worse state. Initially, we want to allow all transitions; then prohibit transitions increasing the violation level of highly ranked constraints; then also prohibit the transitions that would increase the violation marks assigned by lower ranked constraints only, and so forth. Finally, the random walker can only move to neighbours that are not worse.

At each moment, uphill jumps much larger than T have a very low probability, and jumps much smaller than T are extremely likely. By equation (4), T *is equal to* the increase in E that has a likelihood of $1/e$. As the increase in E has been now defined as a pair, so will have to be the temperature T: a pair $\langle K_T, t \rangle \in \mathbb{Z} \times \mathbb{R}^+$.

The first element K_T of the pair is an integer, to be called the *domain* of the temperature. The second element t must be a positive real number. If C_K is an existing constraint, then $T = \langle K, t \rangle$ can be interpreted as if the violation level of

[2] For a more detailed analysis of this definition, see Bíró (forthcoming) and Bíró (2005).

constraint C_K were increased by t. Nonetheless, the domain of the temperature can be different from the indices of existing constraints.

Finally, we define the transition probability $P(w \rightarrow w' \mid T)$. As the rule is to assign a higher index to a higher ranked constraint, the first component of T places temperature somewhere relative to the constraint hierarchy. Lexicographic ordering compares adequately some $\|E(w') - E(w)\| = \langle k, C_k(w') - C_k(w)\rangle$ to $T = \langle K, t\rangle$. This is why the following definition reproduces equation (4):[3]

At temperature $T = \langle K_T, t\rangle$, if $\|E(w') - E(w)\| = \langle k, d\rangle$:

$$P(w \rightarrow w') = \begin{cases} 1 & \text{if } d \leq 0 \\ 1 & \text{if } d > 0 \text{ and } k < K_T \\ e^{-d/t} & \text{if } d > 0 \text{ and } k = K_T \\ 0 & \text{if } d > 0 \text{ and } k > K_T \end{cases} \tag{6}$$

This corresponds to the following *rules of transition*:

- If w' is better than w: move $w \rightarrow w'$!
- If w' loses due to fatal constraint $C_k > K_T$: don't move!
- If w' loses due to fatal constraint $C_k < K_T$: move!
- If w' loses due to the constraint $C_k = K_T$: move with probability $e^{-d/t}$.

In the beginning of the simulation, the domain K_T of the temperature will be higher than the index of the highest ranked constraint; similarly, at the end of the simulation, temperature will drop below the lowest ranked constraint. The most straightforward way to proceed is to use a double loop diminishing temperature.

The pseudo-code of *Optimality Theory Simulated Annealing* (OT-SA) can be presented finally (Fig. 3). The parameters of the algorithm are the initial candidate (w_0) from which the simulation is launched, as well as the parameters of the cooling schedule: K_{max}, K_{min}, K_{step}, t_{max}, t_{min}, t_{step}.

Typically, K_{max} is higher than the index of the highest ranked constraint, in order to introduce an initial phase to the simulation when the random walker may rove unhindered in the search space, and increase even the violation marks assigned by the highest ranked constraint. Similarly, the role of K_{min} is to define the length of the final phase of the simulation. By having K_{min} (much) below the domain (the index) of the lowest ranked constraint, the system is given enough time to "relax", to reach the closest local optimum, that is the bottom of the valley in which the system is stuck. Without such a final phase, the system will return any candidate, not only local optima, yielding an uninteresting model.

[3] As noted by an anonymous reviewer, a major difference between classical SA and SA-OT is that by equation (4), any increase in E has a small theoretical chance of being accepted in classical SA. Yet, SA-OT minimises not a real valued function, but a vector valued function for lexicographical order, due to the Strict Domination Hypothesis. Thus, the vague statement in classical OT that "if $\Delta E \gg T$ then $P \approx 0$" can and has to be formulated here in a more exact way as "if $k > K_T$ then $P = 0$". See Bíró (forthcoming) for further differences between classical SA and SA-OT.

```
ALGORITHM: Simulated Annealing for Optimality Theory
Paramters: w_0, K_max, K_min, K_step, t_max, t_min, t_step
w <-- w_0
   for K = K_max to K_min step K_step
        for t = t_max to t_min step t_step
             choose random w' in neighbourhood(w)
             calculate < C , d >  = ||E(w')-E(w)||
             if d <= 0 then w <-- w'
             else           w <-- w' with probability
                        P(C,d) = 1         , if C < K
                               = exp(-d/t) , if C = K
                               = 0         , if C > K
        end-for
   end-for
return w
```

Figure 3: The algorithm of *Simulated Annealing Optimality Theory* (SA-OT).

Although other options are also possible, the way we shall proceed is placing our n constraints into the domains $K = 0$, $K = 1$,..., $K = n - 1$. That is, the highest ranked constraint receives index $n - 1$, and the lowest one is associated with index 0. Furthermore, $K_{max} = n$ and $K_{step} = 1$.

The parameters t_{max}, t_{min} and t_{step} drive the inner loop of the algorithm, that is, the decreasing of the second component t of temperature $T = \langle K, t \rangle$. This component plays a role only in the expression $e^{-d/t}$, used if the temperature is in the domain of the fatal constraint. Because the neighbouring candidates w and w' typically differ only minimally—a *basic operation* transforms w into w'—, their violation profiles are also similar, thus the difference d in violating the fatal constraint is expected to be low (usually $|d| = 1, 2$). Consequently, $e^{-d/t}$ vanishes if $t \gg 3$, and so the default values used will be $t_{max} = 3$ and $t_{min} = 0$.

The most interesting parameter is t_{step}, for it is inversely proportional to the number of iterations performed (if the other parameters are kept unchanged), and thereby it directly controls the speed of the simulation, that is, its precision. Therefore, we will tune this parameter. Other parameters also may change the number of iterations performed, but their effect is more complex, so tuning t_{step} is the most straightforward way to change the number of iterations. We also could introduce a new parameter for the number of repetitions within the core of the inner cycle.

4 Dutch metrical stress

4.1 The empirical data

Schreuder and Gilbers (2004) analyse the influence of speech rate on stress assignment in Dutch, based on laboratory experiments forcing the participants to produce fast speech. For instance, in normal (slow, andante) speech, the compound word *fototoes-*

tel (‘photo camera’) is assigned a primary stress on its first syllable and a secondary stress on its third syllable (*fótotòestel*). However, in fast (allegro) speech, Schreuder and Gilbers observed a stress shift: the secondary stress moved in a number of cases from the third syllable to the fourth one.

The words used in their experiments belong to the following three groups (Types 1-3). No experiment has been performed with type 0 words. In the stress pattern of a word form or a candidate, s always refers to a syllable with a primary or secondary stress, and u refers to an unstressed syllable hereafter.

Type 0: andante: susu, allegro: suus (OO-correspondence to: su+su)
fo.to.toe.stel ’camera’

Type 1: andante: susuu, allegro: suusu (OO-correspondence to: su+suu)
stu.die.toe.la.ge ’study grant’
weg.werp.aan.ste.ker ’disposable lighter’
ka.mer.voor.zit.ter ’chairman of Parliament’

Type 2: andante: usus allegro: suus (OO-correspondence to: usu+s)
per.fec.tio.nist ’perfectionist’
a.me.ri.kaan ’American’
pi.ra.te.rij ’piracy’

Type 3: andante: ssus allegro: suus (OO-correspondence to: s+su+s)
uit.ge.ve.rij ’publisher’
zuid.a.fri.kaans ’South African’
schier.mon.nik.oog name of in island

In slow (andante) speech, these words are pronounced in a way reflecting their inner structure. Types 0, 1 and 3 are compound words, and they keep the stress pattern of their components unchanged (e.g.: fóto+tòestel or stúdie+tòelage). Additionally, most of the examples in types 2 and 3 end in a suffix that must bear stress. Standard literature on OT phonology uses constraint OUTPUT-OUTPUT CORRESPONDENCE to account for these morphologically based phenomena, as we shall explain it soon.

On the other hand, the fast speech (allegro) forms all display the suus pattern, (followed by an unstressed syllable in the five-syllable words of Type 1). This pattern matches best the markedness constraints, reflecting what the easiest is to pronounce. The markedness constraints used in the analysis advanced by Schreuder and Gilbers (2004) originate from the literature on metrical stress, supposing that parts of the syllables are parsed into *metrical feet*. These constraints are FOOT REPULSION (*$\Sigma\Sigma$) punishing adjacent feet without an intervening unparsed syllable, as well as PARSE-σ, punishing unparsed syllables.

Subsequently, Schreuder and Gilbers propose the re-ranking of the constraints OUTPUT-OUTPUT CORRESPONDENCE and *$\Sigma\Sigma$ above a certain speech rate, after discarding the candidate *(fó)to(tòestel)*—a harmonic bound—from the candidate set. Careful speech is faithful to the morphological structure, as in (7), whereas fast speech

optimises for pronunciation ease is (8).

(7) *Slow (andante) speech:*

fototoestel	OO-CORR.	*ΣΣ	PARSE-σ
☞(fóto)(tòestel)		*	
(fóto)toe(stèl)	*!		*

(8) *Fast (allegro) speech:*

fototoestel	*ΣΣ	OO-CORR.	PARSE-σ
(fóto)(tòestel)	*!		
☞(fóto)toe(stèl)		*	*

Yet, this proposal raises few questions. First, fast speech is usually seen rather as a performance phenomenon. If the competence (the knowledge of the language encoded in the brain) of the speaker is not altered, why would one model it with a new grammar? Second, if we still suppose a sudden change in the grammar at a certain speech rate, how can we explain that the fast speech form appears only in some percentage of cases? If the grammar is altered, then the new form should *always* appear, which is not the case. In fact, the difference between the two speech rates is rather a gradual shift in the frequency of two forms, both of which appear in both andante and allegro speech (Table 1 and Schreuder (2005)).

Stochastic Optimality Theory (Boersma and Hayes 2001) can model this phenomenon within one grammar. By adding a random *evaluation noise* to the ranking of the constraints, Stochastic OT allows for the re-ranking of the two constraints proposed by Schreuder and Gilbers (2004). If noise increases with speech rate, the probability of re-ranking the two constraints also grows, without a categorical switch within the grammar. It is unclear, however, why the evaluation noise should be higher in fast speech. Even worse, Stochastic OT cannot account for the different grammatical form / fast speech form-rates for different words, if they are to be explained by the reranking of the same constraints. The rank of the constraints and the evaluation noise may depend on the speech rate, but not on the specific input.

Third, this particular analysis is based on the re-ranking of *two* constraints, which cannot take place in more than 50% of the cases—leading to a false prediction of the model. In the case of two constraints, the probability of rerankig them converges to 0.5, as the evaluation noise (compared to the difference of their ranks) grows to infinity. However, a Stochastic Optimality Theoretic model with more constraints could correctly predict the fast speech form appearing in more than half of the cases. Which constraint should we then add to the model? An alternative would be to change the (unperturbed) ranks of the constraints, instead of increasing the evaluation noise in fast speech: why and how do the ranks of the different constraints change in function of the speech rate?

The advantage of the model to be presented using *Simulated Annealing Optimality Theory* will be manyfold. First, modelling fast speech by speeding up the algorithm is more convincing than postulating the increase in the evaluation noise or changing the

underlying competence model (the OT grammar). More importantly, SA-OT correctly predicts which words are more likely to be pronounced erroneously. Last, the rate of the fast-speech form may exceed 50% in some cases without having to add new constraints.

4.2 Gen and the topology of the search space

Let us apply simulated annealing to stress assignment. The input is a word composed of a number of syllables. The set of candidates corresponding to this input is composed of all possible correct parses of this input. A parse is correct if: it contains the same number of syllable as the input; it contains at least one foot; feet do not overlap; a syllable not parsed into any foot is unstressed; finally, each foot contains one or two syllables, exactly one of which is stressed. Here, we ignore the difference between primary and secondary stress. For a four-syllable word input, possible parses include: u[s]uu, [su]uu, [us]u[s], [s][s][s][s], etc. Brackets represent foot borders; u and s refer to unstressed and stressed syllables, respectively.

Having defined the set of candidates, we now proceed to the topology of the search space. The *neighbours* of a candidate are the candidates reachable in one *basic step*, and a *basic step* is performing exactly one of the following actions:

- Insert a monosyllabic foot: turn an unparsed u into [s].
- Remove a monosyllabic foot: turn [s] into an unparsed u.
- Move one foot border: enlarge a foot by taking an unparsed syllable into a foot, or narrow a foot by taking an unstressed syllable out of a foot.
- Change the head syllable within a bisyllabic foot.

Defining the topology of the search space includes also determining the probability measure according to which one of the neighbours is picked at each step of the simulation. For the sake of ease, we assign equal probability to each neighbour.

The graph in Figure 4 presents the topology of the search space for a three-syllable input. (The candidate set of four-syllable words includes 43 candidates, and is too complex to reproduce here.) The arcs of the graph connect neighbours, and the arrow on an arc points towards the candidate which is more or equal harmonic, with respect to the toy ranking $*\Sigma\Sigma \gg \textsc{Parse}\text{-}\sigma$.

The arrows already bring us from the "horizontal" to the "vertical" structure of the landscape toured by the random walker. An arc on the graph with only one arrow points downhill, whereas an arc with two arrows represents a horizontal move. An arrow from candidate w to candidate w' means that the move from w to w' is possible with a transition probability of 100% during the entire simulation.

Eyeballing the graph, we can point to some phenomena. Candidate [s][s][s] represents a summit, a local maximum. It is also the global maximum, but this fact cannot be seen directly from the graph. The candidate [s]u[s] is a local minimum: the arrows from all its neighbours point towards it. We also find valleys of candidates of equal harmony, situated lower than their surroundings (for instance the one formed by u[su]

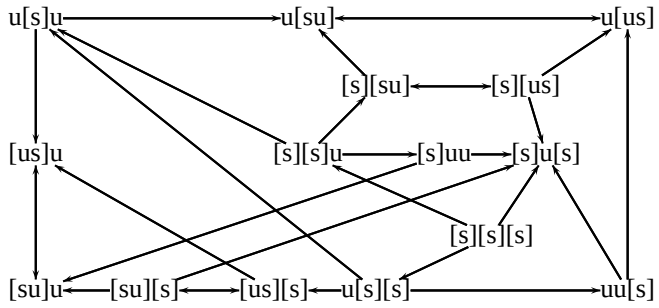


Figure 4: Search space (candidate set, neighbourhood structure) for a three-syllable word.

and u[us]). Comparing the local minima, [s]u[s], u[su], u[us], [us]u and [su]u, proves that all of them are global minima, as well. However, the graph itself would not help in determining which of them is a global minimum.

4.3 The vertical structure of the landscape: the constraints

Besides the constraints $^*\Sigma\Sigma$ and PARSE already mentioned, as well as besides OUTPUT-OUTPUT CORRESPONDENCE to which we are coming back in the next subsection, two further constraints will be used. From the large family of *alignment constraints*, we use the one requiring the left edge of the word matching the left edge of some foot (ALIGN(WORD, FOOT, LEFT), or in short, ALIGN-LEFT). Additionally, constraint TROCHAIC implements the tendency of Dutch to prefer trochaic feet. In sum, here are the constraints we are using:

- ALIGN-LEFT: assign one violation mark if left edge of word does not align with left edge of some foot.
- OUTPUT-OUTPUT CORRESPONDENCE: the stress pattern matches the expectations from the morphological structure.
- $^*\Sigma\Sigma$: one violation mark per adjacent feet borders.
- PARSE: one violation mark per unparsed syllable.
- TROCHAIC: one violation mark to each iambic foot ([us]).

Their ranking should make the grammatical form (the normal speech form) the optimal one, hence faithfulness to the morphological structure should dominate markedness—as it is the case in tableau (7). It is ranking $^*\Sigma\Sigma$ over PARSE which returns suus as the structure preferred by the markedness constraints. ALIGN-LEFT has to be ranked higher than OOC to help suus be a local optimum even for inputs, such as *perfectionist*, whose morphological structure would require usus. Finally, TROCHAIC is ranked low, and its only role is to distinguish between otherwise equal

forms, such as [su]u[su] and [su]u[us] (in a word such as *studietoelage*, whose morphology requires susuu).

In short, without claiming that this is the only possible grammar describing the data, we used the following hierarchy:

(9) ALIGN-LEFT $\gg$ OOC $\gg^{*}$ $\Sigma\Sigma$ $\gg$ PARSE $\gg$ TROCHAIC

We identify constraint ALIGN-LEFT with the domain (index) $K = 4$, constraint OOC with $K = 3$, ..., and finally constraint TROCHAIC with $K = 0$.

4.4 Output-Output Correspondence

In the present subsection, we define the constraint OUTPUT-OUTPUT CORRESPONDENCE (OOC). Originally, Burzio (2002)'s proposal, based on an analogy from physics, required a sum over *all* elements of the lexicon. In practice, however, this constraint compares a candidate with its closest neighbours, that is, with the independent word forms of its morphological constituents. Used to account for phenomena related to morphology, it is usually defined only in a very vague way.

As SA-OT necessitates an exact definition, we propose to define OOC in the following way: candidate w is compared to a string σ of the same length, a stress pattern derived from the stress patterns of w's immediate morphological constituents. If w is the concatenation of a number of morphemes, σ is the concatenation of their stress patterns. Phonological arguments support that a candidate has to be compared to its immediate morphological components, and not to deeper levels in its morphological structure (e.g. Burzio (2002), Bíró (forthcoming)).

For instance, the stress pattern that parses of *individualist* are compared to is $\sigma =$ sususs: the stress pattern susus of *ìndivìduéel* followed by the pattern s of the stress attracting suffix *ist*. The pattern suus of *ìn.di.vi.dú* does not play a role.

After these preparations, we are ready to define the constraint OUTPUT-OUTPUT CORRESPONDENCE. The number of violation marks assigned to a candidate w is the number of mismatches with the corresponding string σ, after a pairwise comparison of the corresponding elements of the (equally long) strings:

(10) $$\mathrm{OOC}_{\sigma}(w) = \sum_{i} \Delta(w_i, \sigma_i)$$

where w_i and σ_i represent the ith letter (now, the ith syllable's type) of the candidate w and the comparison string σ; and where $\Delta(w_i, \sigma_i) = \begin{cases} 1 & \text{if } w_i \neq \sigma_i \\ 0 & \text{if } w_i = \sigma_i \end{cases}$

The definition of OOC is thus complete, but not satisfactory. The result is maybe not exactly what we wish. Misplacing one stress should be a smaller difference than missing a stress entirely, or having extra stresses. If the target string is $\sigma =$ suus, then $w_1 =$ susu should be closer to it than $w_2 =$ suuu or $w_3 =$ suss. Yet, definition (10) will assign two violation marks to w_1, because there is a mismatch in both the third and the fourth syllable, whereas only one violation mark will be assigned to w_2 and to w_3. Candidate w_1 violates constraint OOC_{σ} on the same level as the "totally misconceived" candidate $w_4 =$ssss.

fo.to.toe.stel ‘camera’	*uit.ge.ve.rij* ‘publisher’	*stu.die.toe.la.ge* ‘study grant’	*per.fec.tio.nist* ‘perfectionist’
OOC to: susu	ssus	susuu	usus
fó.to.tòe.stel	*úit.gè.ve.rìj*	*stú.die.tòe.la.ge*	*per.féc.tio.nìst*
fast: *0.82*	fast: *0.65* / **0.67**	fast: *0.55* / **0.38**	fast: *0.49* / **0.13**
slow: *1.00*	slow: *0.97* / **0.96**	slow: *0.96* / **0.81**	slow: *0.91* / **0.20**
fó.to.toe.stèl	*úit.ge.ve.rìj*	*stú.die.toe.là.ge*	*pér.fec.tio.nìst*
fast: *0.18*	fast: *0.35* / **0.33**	fast: *0.45* / **0.62**	fast: *0.39* / **0.87**
slow: *0.00*	slow: *0.03* / **0.04**	slow: *0.04* / **0.19**	slow: *0.07* / **0.80**

Table 1: Simulated (in italics) and observed (in bold; Schreuder, 2005) frequencies. The simulation used $T_{step} = 3$ for fast speech and $T_{step} = 0.1$ for slow speech.

In turn, a modification of the constraint should assign additional violation marks to the difference in the stressed syllables. Let $\| \alpha \|$ denote the number of stresses (s) in the string α: $\| \alpha \| = \sum_i \Delta(\alpha_i, \mathrm{u})$. Then, OOC is re-defined as:

$$\mathrm{OOC}_{z,\sigma}(w) = \sum_i \Delta(w_i, \sigma_i) + z \cdot \Big|\, \| w \| - \| \sigma \| \,\Big| \tag{11}$$

This definition introduces a new parameter z, which determines the relative weight of pointwise mismatch *vs.* difference in the global number of stresses.

4.5 Simulation results

After so much preparation, we can run the simulation. The algorithm of *Simulated Annealing Optimality Theory* has been given in Figure 3. The hierarchy (9) and further considerations mentioned earlier suggest using the following cooling schedule: $K_{max} = 5$ (one layer above the top constraint), $K_{step} = 1$, $t_{max} = 3$, $t_{min} = 0$. Parameter K_{min} was chosen in the function of t_{step}: $K_{min} = -2$ is low enough for $t_{step} = 0.1$ and $K_{min} = -100$ suffices for $t_{step} = 3$.

The simulation has been run with different t_{step} values, ranging between 0.03 and 3. For each parameter setting, we have run the simulation 600 times using each candidate as the initial point of the random walk. Hence, the simulation was run 25800 times for four-syllable inputs (43 candidates), and 71400 times for five-syllable inputs (119 candidates).

The results appear in Table 1, together with the outcome of Maartje Schreuder’s laboratory experiments (Schreuder 2005). Taking $T_{step} = 3$ as a fast speech model, and $T_{step} = 0.1$ as a slow speech model, the match between experiment and simulation is surprisingly good for the words belonging to the type of *uitgeverij*. The quantitative match is worse for other types of words, yet the simulation correctly predicts which types are more likely to be produced erroneously. Furthermore, the results—the 49% of *per.féc.tio.nìst* in fast speech—show that unlike Stochastic OT with the present underlying OT model, SA-OT can return the fast speech form with a frequency above 50% (the difference is significant).

In order to appreciate the results, one has to realise that not only did the model reproduce the grammatical forms, but it also correctly predicted which among the 43 or 119 candidates is the alternative fast speech form. In fact, in the case of *perfectionist*, a third form has also been returned (2% in slow speech, 12% in fast speech), namely [s][su]u (*pérfèctionist*)—by using $z = 1$ in the definition (11) of OOC. Different values for z returned the non-attested [s][su]u form even more frequently. This difficulty underlines the non-triviality of the present results.

5 Summary

The present paper has implemented a heuristic technique, simulated annealing, to Optimality Theory. The standard algorithm had to be slightly modified in order to use it to find the optimal candidate of the candidate set. Simulated annealing does not guarantee maximal precision, and this "drawback" could model the lack of precision in human speech: faster production yields more performance errors. Despite quantitative mismatches so-far, the approach seems to be promising.

Simulated annealing required the introduction of some new concepts in Optimality Theory: a *topology* (a *neighbourhood structure*) on the candidate set, the *difference* of two violation profiles, temperature, as well as a more precise definition of OUTPUT-OUTPUT CORRESPONDENCE.

We propose to see *Simulated Annealing Optimality Theory* (SA-OT) as a model for (part of) the linguistic performance. If traditional Optimality Theory represents linguistic competence (that is, the static knowledge of the language encoded in one's brain), then simulated annealing models the dynamic computations involved in producing utterances. The arguments for why simulated annealing can be an adequate model of (part of) the performance included the fact that it does not require complex computing capacities even in the case of NP-complete problems; that it returns a "nearly good" solution in limited time; and that this time interval can be reduced (just like speech can be speeded up) by paying in precision. The last fact was demonstrated on the case of Dutch stress assignment in fast speech.

The reader is welcome to try out the demo of SA-OT and the implementation of the model introduced for Dutch stress at http://www.let.rug.nl/ birot/sa-ot/.

Acknowlegments

I acknowledge the support of the *High Performance Computing* program of the University of Groningen, the Netherlands. The author is thankful to *Gosse Bouma*, *Dicky Gilbers*, *Gertjan van Noord* and *Maartje Schreuder* for data and for valuable discussions, as well as to an anonymous reviewer for useful remarks.

References

Bíró, T.(2003), Quadratic alignment constraints and finite state optimality theory, *Proc. FSMNLP, within EACL-03, Budapest*, also: ROA-600[4], pp. 119–126.

[4]ROA stands for *Rutgers Optimality Archive* at `http://roa.rutgers.edu/`.

Bíró, T.(2005), How to define Simulated Annealing for Optimality Theory?, *Proc. Formal Grammar 10 and MOL 9*, Edinburgh.

Bíró, T.(forthcoming), *Finding the Right Words: Implementing Optimality Theory*, PhD thesis, Rijksuniversiteit Groningen, Groningen, Netherlands.

Boersma, P. and Hayes, B.(2001), Empirical tests of the gradual learning algorithm, *Linguistic Inquiry* **32**, 45–86.

Burzio, L.(2002), Missing players, *Lingua* **112**, 157–199.

Eisner, J.(2000), Easy and hard constraint ranking in OT, *Finite-State Phonology: Proc. SIGPHON-5*, Luxembourg, pp. 57–67.

Howells, T.(1988), Vital: a connectionist parser, *Proceedings of 10th Annual Meeting of the Cognitive Science Society*, pp. 18–25.

Kempen, G. and Vosse, T.(1989), Incremental syntactic tree formation in human sentence processing, *Connection Science* **1**, 273–290.

Kirkpatrick, S., Jr., C. D. G. and Vecchi, M. P.(1983), Optimization by simulated annealing, *Science* **220**(4598), 671–680.

Kuhn, J.(2000), Processing optimality-theoretic syntax by interleaved chart parsing and generation, *Proc.ACL-38*, *Hongkong*, pp. 360–367.

Prince, A. and Smolensky, P.(2004), *Optimality Theory: Constraint Interaction in Generative Grammar*, Blackwell, Originally: RuCCS-TR-2, 1993.

Reeves, C. R. (ed.)(1995), *Modern Heuristic Techniques for Combinatorial Problems*, McGraw-Hill, London, etc.

Schreuder, M.(2005), *Prosodic Processes in Language and Music*, PhD thesis, Rijksuniversiteit Groningen, Groningen, Netherlands.

Schreuder, M. and Gilbers, D.(2004), The influence of speech rate on rhythm patterns, *in* D. Gilbers, M. Schreuder and N. Knevel (eds), *On the Bounderies of Phonology and Phonetics*, University of Groningen, pp. 183–201.

Selman, B. and Hirst, G.(1994), Parsing as an energy minimization problem, *in* G. Adriaens and U. Hahn (eds), *Parallel Natural Language Processing*, Ablex, Norwood, NJ, pp. 238–254.

Smolensky, P.(1986), Information processing in dynamical systems: Foundations of harmony theory, *Rumelhart et al.: Parallel Distributed Processing*, Vol. 1, Bradford, MIT Press, Cambridge, London, pp. 194–281.

Tesar, B. and Smolensky, P.(2000), *Learnability in Optimality Theory*, The MIT Press, Cambridge, MA - London, England.

Query-Based Summarization using Rhetorical Structure Theory

Wauter Bosma

Human Media Interaction, University of Twente

Abstract

Research on Question Answering is focused mainly on classifying the question type and finding the answer. Presenting the answer in a way that suits the user's needs has received little attention. This paper shows how existing question answering systems—which aim at finding precise answers to questions—can be improved by exploiting summarization techniques to extract more than just the answer from the document in which the answer resides. This is done using a graph search algorithm which searches for relevant sentences in the discourse structure, which is represented as a graph. The Rhetorical Structure Theory (RST) is used to create a graph representation of a text document. The output is an extensive answer, which not only answers the question, but also gives the user an opportunity to assess the accuracy of the answer (is this what I am looking for?), and to find additional information that is related to the question, and which may satisfy an information need. This has been implemented in a working multimodal question answering system where it operates with two independently developed question answering modules.

1 Introduction

A question answering (QA) system pinpoints an *answer* to a given question in a set of documents. A *response* is then generated for this answer, and presented to the user (c.f. Hirschman and Gaizauskas 2001). Discussion of the task of pinpointing the answer is beyond the scope of this paper. I will assume that the sentence which best matches the question, the *answer sentence*, is located by a QA system in a corpus of text documents. What remains is the task of generating an appropriate response and present it to the user.

Question answering systems traditionally try to find an 'exact answer'. An exact answer is a "text string consisting of a complete answer and nothing else" (Voorhees 2003). Strings that contain a correct answer with additional text are considered 'inexact'. Finding exact answers is also the focus of large-scale question answering evaluation programs such as TREC (Voorhees and Tice 2000).

Studies have shown, however, that users appreciate receiving more information than *only* the exact answer (Burger et al. 2000). Consulting a question answering system is only part of a user's attempt to fulfill an information need: it's not the end point, but some steps along what has been called a 'berry picking' process, where each answer/result returned by the system may motivate a follow-up step (Bates 1990). The user may not only be interested in the answer to the question, but also in related information. The 'exact answer approach' fails to show leads to related information that might also be of interest to the user. Lin et al. (2003) show that when searching for information, increasing the amount of text returned to users significantly decreases the number of queries that they pose to the system, suggesting that users utilize related information from supporting text.

In both commercial and academic QA systems, the response to a question tends to be more than the exact answer, but the sophistication of their responses varies from system to system. There are three degrees of sophistication in response generation.

Exact answer. The most basic form of answer presentation is to present only an exact answer. For instance, an exact answer to the question "what is the cause of RSI?" could be:

the movement always involves contraction of the same muscles

Answer plus context. If only an exact answer is provided, users have great difficulty assessing the accuracy of the answer, and thus whether the answer is correct. If the user is provided with more context (i.e. surrounding text), she will exploit this in order to find out whether the answer is indeed an answer to the question (Lin et al. 2003). Most of the current QA systems follow this approach, and return not only the answer but also part of the surrounding text, in which the answer itself may be highlighted. This can be a few lines of text, or only the single sentence in which the answer occurs. For instance, the response to the question about RSI causes could consist of the answer sentence, the preceding sentence and the sentence following the answer sentence:

Despite fewer working hours, the same quantity of work had to be finished. A possible explanation of the development of RSI as a result of frequently repeated movements which are performed with low exertion is that **the movement always involves contraction of the same muscles**. *This happens for instance when working with a display device.*

Extensive answer. Lin et al. (2003) have shown that users prefer to receive more information than only an exact answer, but simply returning to the user a particular quantity of surrounding text is likely to produce incoherent results. Furthermore, the surrounding text may include irrelevant information or unnecessary details. Although—similarly to an answer plus context—an extensive answer includes more information than just the exact answer, the difference is that the extensive answer approach specifically aims at producing a coherent response that includes, apart from the answer, also related information which might interest the user. For instance, an extensive answer to the question about RSI causes could be:

A possible explanation of the development of RSI as a result of frequently repeated movements which are performed with low exertion is that **the movement always involves contraction of the same muscles**. *This happens for instance when working with a display device. Eventually they can cease to function and the muscle will lose strength.*

This paper presents a method to produce extensive answers by extracting the sentences which are most salient with respect to the question, from the document which contains the answer. This is very similar to creating an extractive summarization: in both cases, the goal is to extract the most salient sentences from a document. In case

of summarization, the result should reflect the communicative intent conveyed by the original document, i.e. the summarization contains the most salient parts of the original document. In question answering, what is relevant depends on the user's question rather than on the intention of the writer of the document which happens to contain the answer. In other words, the output of the summarization process is adapted to suit the user's declared information need (i.e. the question). This branch of summarization has been called *query-based summarization* (c.f. Chali 2002).

The method proposed here uses a pointer to the (exact) answer as a summarization parameter. The sentences which are most closely related to the answer sentence are extracted and the resulting extensive answer is presented to the user. This answer includes the answer sentence itself. For this type of summarization, determining the salience of a sentence as done in generic summarization no longer suffices. Instead of using a static notion of salience, the strength of the relation between the answer and each sentence is used for summarization. Rhetorical Structure Theory is used to find those relations.

In short, the following method is proposed. The rhetorical (RST) structure of the document to be summarized is transformed into a weighted graph, in which each vertex represents a sentence. The weight of an edge represents the distance between the two sentences. Given that a sentence a is relevant to the answer, the weight of a path from sentence a to another sentence b represents the level of relevance of sentence b to the answer. Given an appropriate assignment of weights in the graph, such a graph can be used to determine which sentences are the most relevant to the answer.

This paper is structured as follows. First, background knowledge about coherence, Rhetorical Structure Theory and summarization is provided in section 2. Section 3 discusses the proposal to answer extension and section 4 discusses its application in a real system. This paper concludes with a discussion and possible follow-ups on this research in section 5. Although this work is aimed at the Dutch language, all examples have been translated to English. This is possible because all methods presented in this paper are language independent.

2 Background

2.1 Coherence in Discourse

What makes discourse different from just any list of sentences, is that sentences in discourse are somehow related to each other, i.e. by means of coreference, substitution, ellipsis, conjunction and lexical cohesion (Halliday and Hasan 1976). All these phenomena account for relations between words or groups of words sentences in discourse. Such relations are called *cohesive* relations (Mani, Bloedorn and Gates 1998).

However, it is argued that there is more to discourse than only cohesion. Several theories have been developed to model the structure of discourse, most notably the intentional structure of Grosz and Sidner (1986) and the rhetorical structure (RST) of Mann and Thompson (1987). Both theories state that discourse can be segmented into non-overlapping spans of texts, that an intentional relation holds between those segments, and that a segment may in turn be further segmented into smaller segments which are also subject to an intentional relation.

The main difference between theories of text organization is the number of relation types that can be identified. Some argue that any coherence relation between two spans of text can be classified as one of a finite number (usually in the order of tenths) of rhetorical relation types (c.f. Mann and Thompson 1988). Others state that the number of possible rhetorical relations is ultimately infinite, so it makes no sense trying to classify relations or to define a definite relation set (c.f. Grosz and Sidner 1986). Instead, Grosz and Sidner (1986) restrict themselves to only two relations—DOMINANCE and SATISFACTION-PRECEDENCE.

2.2 Rhetorical Structure Theory

For the purpose of text summarization, RST has theoretical and pragmatic advantages over other theories. Good levels of agreement have been measured between human annotators of RST, which indicates that RST is well defined (Mann and Thompson 1988, den Ouden 2004). Furthermore, a corpus of RST-annotated English news articles is publically available, which can be used for training and evaluating RST-based summarization algorithms (Carlson, Marcu and Okurowski 2002). Another advantage of RST is that RST defines coherence relations very formally and elaborately, which makes computational applications easier to develop.

According to RST, a rhetorical relation typically holds between two contiguous spans, of which one span (the *nucleus*) is more central to the writer's intention that the other (the *satellite*), whose sole purpose is to increase the reader's understanding or belief of what is said in the nucleus. Sometimes, two related spans are of equal importance, in which case there is a *multinuclear* relation between them. The related spans form a new span, which can in turn participate in a relation with another span. The smallest units of discourse are *elementary discourse units* or *edus*.

The idea behind RST is that all rhetorical relations that can possibly occur in a text can be categorized into a finite set of relation types. The Rhetorical Structure Theory is primarily a method of text analysis. Mann and Thompson (1988) define a set of discourse relations that commonly occur in English texts, but RST has also been applied with other relation sets (such as in Carlson and Marcu 2001). The optimal relation set may depend on the genre and the application (Marcu and Echihabi 2002, André and Rist 1995)

2.3 Query-based Summarization

There are several flavors of summarization:

Abstractive vs. extractive. A feature of an extractive summarization is that each sen-

tence of the summarization is literally copied from the source document. Abstracting involves *rewriting* a text in fewer words, rather than *extracting* the most salient portions of a text.

Multi-document vs. single-document. A multi-document summarization contains the most relevant information from a set of documents, while in single-document summarization, only a single document is used.

Query-based vs. generic. A query-based summarization is tailored to suit the user's declared information need, while a generic summarization reflects the writer's communicative intent as conveyed by the source document.

This paper discusses query-based single-document extracts—the summarization will not contain any sentences that are not present in the original document. The query is a question posed by the user. Because the answer is already pinpointed in a document by a question answering engine, a pointer to the answer can be used as a summarization parameter.

While creating an extract for a particular answer, a candidate sentence can only be included if something is known about the relation between the candidate sentence and the answer sentence. Indications of a strong relation between two sentences include statistical measures of text similarity, such as the number of denotations of mutually used concepts. This paper focuses on the use of rhetorical relations. More in particular, RST.

RST has proven to be very useful to facilitate summarization (Marcu 1997). In his summarization effort, Marcu used the nuclearity of relations in the rhetorical structure to determine which sentence is more salient, but he also explored other features as additional indicators of importance, such as sentence length (Marcu 1997, Marcu 1998).

The elementary discourse units of the RST analyses used for summarization are sentences. RST can be used to make a more detailed analysis of discourse, including relations between clauses, but for making an extractive summarization, using a finer granularity than sentences is not necessary. If more detailed analyses were used, the extract could also contain parts of sentences, but this would require rewriting the extracted text into a grammatical whole.

Query-based summarization has been applied in information retrieval (c.f. Chali 2002, Saggion, Bontcheva and Cunningham 2003), but also in multi-document summarization (Mani and Bloedorn 1997). In multi-document summarization—like in question answering—the source documents of the summarization are not written to satisfy the information need expressed by the query at hand.

Mani and Bloedorn (1997) used graphs to formalize relations between sentences inside a document for multi-document summarization. A spreading activation algorithm is then used to perform a query-based summarization, given a starting node that is selected for the query. Although Mani and Bloedorn (1997) aim at multi-document summarization, a similar graph-based algorithm to perform query-based summarization can also be applied in single-document summarization, as demonstrated by this paper.

3 An Approach to Query-Based Summarization Using RST

This section describes a two-step approach to query-based summarization. First, the relations between sentences are defined in a discourse graph. Then, this graph is used to perform the summarization. During the first step, the rhetorical structure is transformed into a graph representation. The second step exploits a graph search algorithm in order to extract the most salient sentences from the graph. The starting node of the search is the node representing the answer sentence.

The summarization should consist of the most salient sentences, given the starting node. This can be realized by determining the *distance* between the answer sentence and each of the other sentences. The sentences which are most closely related to the answer sentence are included in the summarization.

A simple measure of distance between two sentences would be the *linear distance*, i.e. the number of sentences in between the two sentences, given the linear order of the sentences in the text. For instance, a summarization could consist of the answer sentence and a number of successive (and/or preceding) sentences. However, experience shows that summarizations are often incoherent if they are based on solely this measure of distance between sentences. At paragraph boundaries, for instance, two contiguous sentences can be rhetorically very distant.

The distance between sentences can also be measured by their distance in the RST graph, which I call the *rhetorical distance*. The Rhetorical Structure Theory defines relations between two spans of text, which can be used to derive the distance from one sentence to another. The graph which is created from the rhetorical structure can be used as a computational model for summarization.

The most nuclear sentence of an RST analysis is the sentence which is most central to the writer's purpose. The graph ensures that, similarly to Marcu's approach, a nucleus is preferred over a satellite: in both summarization approaches, a satellite cannot be included in a generic summarization without its nucleus. The consequence is that in the specific case that the entry point of the summarization—the answer sentence—is the most nuclear sentence in the RST analysis, the result resembles the result of the summarization approach by Marcu (1997). However, the graph-based approach is more general in the sense that the summarization can start from any specific sentence rather than only the most nuclear sentence of the analysis.

RST analyses as weighted graphs

It is relatively straightforward to derive a graph from a rhetorical structure. While RST is not designed as a computational framework, graph theory is very suited for this purpose. A RST tree can be converted to a discourse graph by means of the following steps.

1. For each elementary discourse unit in the RST tree, create a vertex associated with it.

2. For each directed relation, create an edge from the nuclear sentences of the nucleus to the nuclear sentences of the satellite of the relation.

Figure 1: Rhetorical structure examples.

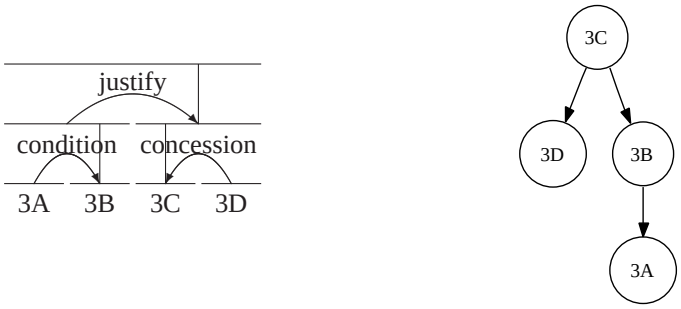


Figure 2: Rhetorical structure example and a discourse graph created for this rhetorical structure.

A sentence is a nuclear sentence of a text span if it is not part of any sub span (of the text span) which participates as a satellite in a directed relation with any other sub span. A text span can have multiple nuclear sentences if multinuclear relations are involved. For instance, in the RST diagram on the left in Figure 1, the set of nuclear sentences of the entire document (denoted as 1A:1D) contains only sentence 1C. The right diagram shows a rhetorical structure in which the set of nuclear sentences of 2A:2D consists of sentences 2C and 2D.

The result of the transformation is an a-cyclic directed graph of which the vertices correspond to elementary discourse units, and the edges define relations between them. Figure 2 shows an example of a rhetorical structure and a discourse graph that was created as described above. During the transformation from RST to graph, part of the structural information is lost because sentences of the graph are directly connected to other sentences, while in RST, one end of a relation can also span more than one discourse unit. If in RST one sentence was related to a text span of two sentences, it is related to the nucleus of the two sentences in the discourse graph. In practice, this means that if the inclusion of a sentence in a summarization was justified by a rhetorical relation, the nucleus of that relation must be included in the summarization as well. This is in line with Mann and Thompson's (1988) definition of directedness

Figure 3: Rhetorical structure containing a multinuclear relation and the corresponding discourse graph.

of relations, which states that a nucleus of a directed relation has meaning without the satellite, but not the other way round.

If a multinuclear relation is involved, as in Figure 3, each of the sentences participating in the multinuclear relation (in the example: sentences 4B, 4C and 4D) is connected with the nucleus of the multinuclear span. That is, in the example, sentence 4A is connected to each of the sentences 4B, 4C and 4D, but sentences 4B–4D are not directly mutually connected. The reason for this is that in terms of RST, there is a mutual (multinuclear) relation between the sentences 4B–4D, but only in the context of this relation. They are mutually independent: if we know that 4B contains relevant information in a particular context, there is no way to be sure that, to any extent, 4C is relevant as well, based on the relevance of 4B.

Now we have a discourse graph T, we assume that given two sentences $a, b \in T$ for which there is a path from a to b, we can say that they are related and therefore if a is relevant to the answer, b is also relevant to the answer. If a path contains more than one edge, the sentences are related only indirectly and an indirect relation is weaker than a direct relation between two sentences.

The strength of a relation between two sentences could be calculated by just counting the number of edges in the path between the vertices of the sentences. However, it may be the case that there is more than one sentence with an equally long path to the starting point of the summarization. This means that during a summarization, the two sentences are equally likely to be included in the summarization, although there may be other indications of one sentence being better suited for inclusion in the summarization than the other.

In order to remedy this situation, we can assign weights to vertices and edges in the discourse graph. A greater distance is reflected by a greater weight. A low weight of the path from a to b indicates a high probability that b is relevant given, that a is relevant. The total weight of the path from a to b is denoted as $weight(a, b)$. The weight of a path between two sentences is defined as long as if there is a path that connects them. The weight of a path is the sum of the weights of its edges and vertices.

Given the entry point of the summarization (the answer sentence), the shortest path from the this sentence to any other sentence defines the relevance of the topic of the

other sentence to the final answer. All we have to do now in order to be able to extract an answer, is to determine the weights.

3.1 Determining Weights

Weights of edges in the discourse graph can be determined by using features of the rhetorical structure from which the graph was created, such as features of the text spans on either side of the relation for which the edge was created as well as features of the relation itself. Also vertices can be weighted. The weight of a vertex depends on features of the sentence it corresponds to. The only constraint is that all weights of edges and vertices are non-negative.

The rhetorical structure has many features that may be relevant for determining weights to edges or vertices. Currently, only three features are considered when assigning weights. For these features, there is at least some evidence that they can contribute to the quality of a summarization. Further research may motivate the use of other features as well. For instance, the algorithm does not differentiate between relation types because there is not sufficiently specific evidence to support this. The following features are considered, in order of relative importance.

1. Each edge has a basic weight, which is the same for all edges in the graph. This makes the distinction between directly and indirectly related sentences explicit. Two sentences are less closely related if the path that connects them consists of more edges.

2. For each edge, a weight is added depending on the number of sentences in the satellite of the corresponding rhetorical relation. If a particular satellite contains more sentences than another satellite of the same nucleus, the author apparently spent more words on it, which may indicate that the author finds this topic more important than a shorter one, although they both are a satellite of the same nucleus.

3. For each vertex, a weight is added depending on the number of words in the sentence. According to Marcu (1998), this is a good measure for the amount of new information contained in the sentence.

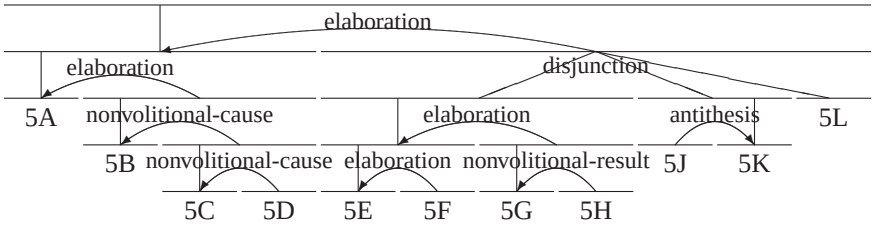


Figure 4: Rhetorical structure tree of the text fragment.

The weights of edges and vertices are calculated as follows.

$weight(e) =$

$a + b \cdot \frac{1}{sentences(sat(r))}$, if e is the edge that was created for the relation r, where $sat(r)$ is the satellite of r, and $sentences(s)$ is defined as the number of sentences of a span s, a is the basic weight, and b is a constant factor of the 'satellite size' component of the edge weight;

$weight(v) =$

$c \cdot \frac{1}{words(s)}$, if v is the vertex that was created for the sentence s, where $words(s)$ is the number of words in s, and c is a constant.

The constants a, b and c are used to balance the three factors of the distance between two sentences: the number of edges (represented by a) is more important than the number of sentences in the satellite (represented by b), and the number of sentences in the satellite is more important than the number of words in the sentence (represented by c).

Example 1: Extraction

This example shows how three sentences can be extracted from a text, based on its RST analysis, and given the entry point of the summarization. In a QA context, the entry point would be the answer sentence. Two of the extracted sentences are direct or indirect satellites of the answer sentence, the third is the answer sentence itself. The RST analysis of the following (segmented) text is shown in Figure 4. The entry point for the extraction is sentence 5E.

[A high pressure of workload, stress and repeatedly carrying out the same operation for a long period of time are the most important factors causing RSI to develop.][5.A]

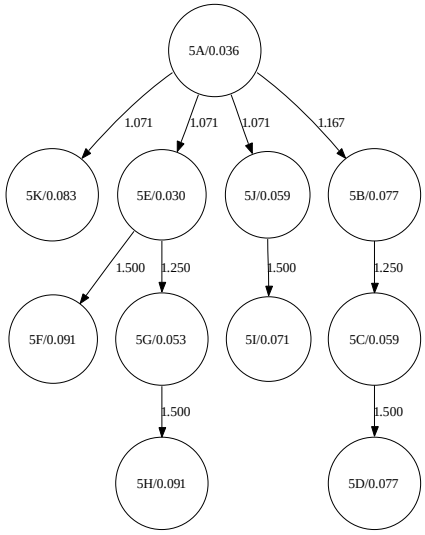


Figure 5: Weighted rhetorical structure graph of a text fragment. The vertices are labeled *sentence*/*weight*, in which *sentence* refers to the sentence corresponding to the vertex. The edges are labeled by their weights.

[In the Netherlands the work pressure increased with approximately 1.5% per year.]^{5B} [This is the result of shorter working hours in the eighties and nineties of the twentieth century.]^{5C} [Despite fewer working hours, the same quantity of work had to be finished.]^{5D} [A possible explanation of the development of RSI as a result of frequently repeated movements which are performed with low exertion is that the movement always involves contraction of the same muscles.]^{5E} [This happens for instance when working with a display device.]^{5F} [The motorial entities can be damaged because of oxygen lack and the impossibility of removing waste products.]^{5G} [Eventually they can cease to function and the muscle will lose strength.]^{5H} [There are however also indications that the complaints do not arise from damaged muscles.]^{5J} [Instead, they supposedly arise from abnormalities in the response of the brain to signals from the muscles.]^{5K} [Another possibility is that psychological factors can lead to symptoms of RSI.]^{5L}

First, a discourse graph is created from an RST analysis (as shown in Figure 5). The graph contains weighted edges and vertices. For this graph, the total weight of the paths from sentence 5E to each sentence in the graph is calculated using Dijkstra's shortest paths algorithm (Dijkstra 1959). A path in a graph is an alternating sequence of vertices and edges, beginning and ending with a vertex. For instance, in the graph of Figure 5, there is a path over three vertices and two edges from 5E to 5H. The weight of this path is the sum of the weights of all of its edges and vertices. In the

	5A	5B	5C	5D	5E	5F	5G	5H	5J	5K	5L
			...					...			
5E	—	—	—	—	0.030	1.621	1.333	2.924	—	—	—
			...					...			

Table 1: Weight table showing the total weight of the path from 5E to each sentence in the rhetorical structure graph of Figure 5.

Figure 6: Extraction graph of the three sentences selected for inclusion in the summary, and the corresponding structure in RST notation, which is derived from the original RST analysis.

case of the path from 5E to 5H, this is $0.03 + 1.25 + 0.053 + 1.5 + 0.091 = 2.924$.

The weights of the paths originating from 5E are shown in Table 1. Only four sentences are reachable from 5E. Since the selection of sentences is based on the weight of their path from 5E, a sentence which is associated with an unreachable vertex cannot be included in the extract.

From this table, the sentences with the cheapest path from the entry point 5E are selected. The selected sentences are filtered out, resulting in the discourse graph on the left in Figure 6. For the sentences in this graph, the rhetorical structure can be derived using the original RST analysis in Figure 4. The result is the rhetorical structure in Figure 6. This rhetorical structure may be used for further processing, for example for the purpose of speech synthesis (den Ouden 2004). The output of the extraction process would be the following text. The answer sentence is highlighted.

> **A possible explanation of the development of RSI as a result of frequently repeated movements which are performed with low exertion is that the movement always involves contraction of the same muscles.** This happens for instance when working with a display device. The motorial entities can be damaged because of oxygen lack and the impossibility of removing waste products.

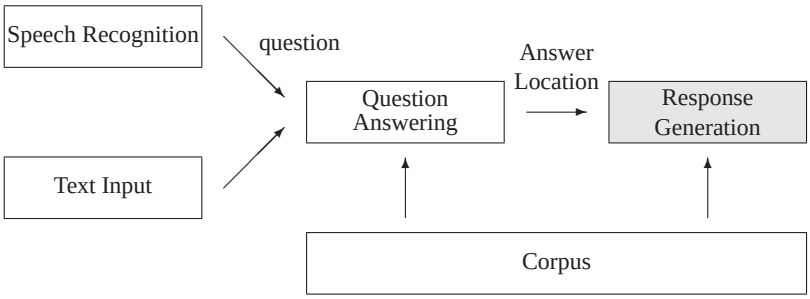


Figure 7: Simplified architecture of the IMIX system. The work in this paper is implemented in the 'response generation' module.

4 Answer Extraction in IMIX

The approach to query-based summarization is implemented as part of a working multimodal question answering system, which has been developed within the context of IMIX. IMIX is the Interactive Multimodal Information Extraction program of the Netherlands Organization for Scientific Research (NWO), with the objective of building a fully multimodal question answering dialog system (i.e. multimodal input and output). Currently, there is a first version of the IMIX system which is capable of answering typed and spoken questions in Dutch about medical issues. The answer is presented using speech, and an HTML page with text and images. Other IMIX modules are responsible for question answering, speech recognition, speech synthesis and the graphical user interface.

A simplified model of the architecture of the IMIX system is depicted in Figure 7. The Question Answering module receives a spoken or typed question from Speech Recognition or Text Input. The output of Question Answering is a pointer to a single sentence in a corpus, which is shared between Question Answering and Response Generation. This paper describes the 'Response Generation' module, which takes the question answering result (the answer sentence) as input for producing a coherent response. The Response Generation module has access to the QA corpus. Therefore, it has access to not only the sentence that was found by QA, but also to its context, i.e. to the entire document in which the answer sentence resides.

The response generation module in IMIX uses the summarization method described in this paper. Because in IMIX the system's response to questions has to be brief, the size of the responses is limited to a maximum of three sentences. The generated responses have not yet been formally evaluated, but information evaluations show that the responses are generally coherent, and that additional sentences (beyond the answer sentence) contain information which is strongly related to the question. The following are examples of responses that were generated for questions by the IMIX system.

Question: *What is RSI?*

Answer: RSI is a name for a large number of diseases which affect the neck, shoulders, arms and hands. *Repetitively making the same movements may cause complaints.*

Question: *What is the cause of RSI?*

Answer: A possible explanation of the development of RSI as a result of frequently repeated movements which are performed with low exertion is that the movement always involves contraction of the same muscles. *This happens for instance when working with a display device. Eventually they can cease to function and the muscle will lose strength.*

Although automated RST analysis can be performed on English texts (Marcu and Echihabi 2002), this is not yet the case for Dutch. Because Dutch is the interaction language of IMIX, the RST analyses used for extraction still have to be created manually. Because this is very time-consuming, at present, the RST-analyzed corpus is only a subset of the QA corpus. In cases where an RST analysis is missing, the response generation module falls back to giving only the answer sentence instead of a multi-sentence extract.

5 Discussion and Future Work

Question answering systems can benefit from responding with more extensive answers by means of query-based summarization. The presented approach to query-based summarization consists of two steps. First, the rhetorical structure tree is used to build distance graphs which determine the distances between individual sentences. Then, these graphs are used to decide which sentences are most relevant to the answer. The result is an answer that is more informative than an 'exact answer' (as returned by traditional QA systems), and more concise than a full document (as returned by IR systems)—a compromise between question answering and information retrieval, taking the best features from both.

The advantage of the separation between formalization (graph construction) and extraction (graph search and sentence extraction) is that the latter is fairly generic: it can also be applied to discourse graphs that are not RST-based. Mani and Bloedorn (1997) experimented with summarization based on conceptual similarity relations between sentences. The conceptual graphs could be integrated with the RST-based graphs, in order to exploit all available indications of relevance.

The extraction method has been tested with promising results on a limited scale in the IMIX question answering system, but more thorough experiments are required in order to test both the performance of the approach and the validity of the more general case of extending answers using the source document.

Future versions of the IMIX system will be capable of participating in more complex dialogs than just answering isolated questions. Because the summarizer is aware of coherence relations, its output is also RST-annotated text. Being able to reason

about its output is very useful for a dialog system in order to parse and reply to subsequent utterances of the user. For instance, it would be useful for a dialog system to know that part of its output participates in an 'evidence' relation with another portion of the output. RST can also be used to improve speech synthesis (den Ouden 2004).

Another challenge is to investigate how query-based summarization methods apply to multimodal documents. Rhetorical Structure Theory itself applies to multimodal documents without any extensive modifications (c.f. André 1994), but this direction of RST has to be further explored, and further tools have to be developed for the generation of multimedia responses including pictures and animations.

Acknowledgements

This work is funded by the Interactive Multimodal Information Extraction (IMIX) program of the Netherlands Organization for Scientific Research (NWO).

References

André, E. (1994), *Ein plan-basierter Ansatz zur Generierung multimedialer Präsentationen*, PhD thesis, Universität des Saarlandes, Saarbrücken.

André, E. and Rist, T. (1995), Generating coherent presentations employing textual and visual material, *Artificial Intelligence Review, Special Volume on the Integration of Natural Language and Vision Processing* **9**(2–3), 147–165.

Bates, M. J. (1990), The berry-picking search: user interface design, *in* H. Thimbleby (ed.), *User Interface Design*, Addison-Wesley.

Burger, J., Cardie, C., Chaudhri, V., Gaizauskas, R., Harabagiu, S., Israel, D., Jacquemin, C., Lin, C.-Y., Maiorano, S., Miller, G., Moldovan, D., Ogden, B., Prager, J., Riloff, E., Singhal, A., Shrihari, R., Strzalkowski, T., Voorhees, E. and Weishedel, R. (2000), Issues, tasks, and program structures to roadmap research in question & answering (q&a), NIST DUC Vision and Roadmap Documents.

Carlson, L. and Marcu, D. (2001), Discourse tagging manual, *ISI Tech Report ISI-TR-545*, Information Sciences Institute.

Carlson, L., Marcu, D. and Okurowski, M. E. (2002), Building a discourse-tagged corpus in the framework of rhetorical structure theory, *in* J. van Kuppevelt and R. Smith (eds), *Current Directions in Discourse and Dialogue*.

Chali, Y. (2002), Generic and query-based text summarization using lexical cohesion, *in* R. Cohen and B. Spencer (eds), *Advances in Artificial Intelligence: 15th Conference of the Canadian Society for Computational Studies of Intelligence, AI 2002*, Calgary, Canada, pp. 293–302.

den Ouden, H. (2004), *Prosodic realizations of text structure*, PhD thesis, University of Tilburg.

Dijkstra, E. (1959), A note on two problems in connection with graphs, *Numerische Mathematik* **1**, 269–271.

Grosz, B. J. and Sidner, C. L. (1986), Attention, intentions and the structure of discourse, *Computational Linguistics* **12**(3), 175–204.

Halliday, M. A. and Hasan, R. (1976), *Cohesion in English*, Longman, London.

Hirschman, L. and Gaizauskas, R. (2001), Natural language question answering: The view from here, *Natural Language Engineering* **7**(4), 275–300.

Lin, J., Quan, D., Sinha, V., Bakshi, K., Huynh, D., Katz, B., and Karger, D. R. (2003), What makes a good answer? the role of context in question answering, *Proceedings of the Ninth IFIP TC13 International Conference on Human-Computer Interaction*, Zürich, Switzerland.

Mani, I. and Bloedorn, E. (1997), Multi-document summarization by graph search and matching, *Proceedings of the Fourteenth National Conference on Artificial Intelligence (AAAI'97)*, pp. 622–628.

Mani, I., Bloedorn, E. and Gates, B. (1998), Using cohesion and coherence models for text summarization, *Proceedings of AAAI Spring Symposium on Intelligent Text Summarization*, Stanford.

Mann, W. C. and Thompson, S. A. (1987), Rhetorical structure theory: A theory of text organization, *Technical Report ISI/RS-87-190, NTIS Identifying Number ADA 183038*, University of Southern California, Information Sciences Institute, Marina del Rey, CA, United States.

Mann, W. C. and Thompson, S. A. (1988), Rhetorical structure theory: Toward a functional theory of text organization, *Text* **8**(3), 243–281.

Marcu, D. (1997), From discourse structures to text summaries, *The Proceedings of the ACL'97/EACL'97 Workshop on Intelligent Scalable Text Summarization*, Madrid, Spain, pp. 82–88.

Marcu, D. (1998), To build text summaries of high quality, nuclearity is not sufficient, *The Working Notes of the the AAAI-98 Spring Symposium on Intelligent Text Summarization*, AAAI, Stanford, CA, pp. 1–8.

Marcu, D. and Echihabi, A. (2002), An unsupervised approach to recognizing discourse relations, *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL-2002)*, Philadelphia, PA.

Saggion, H., Bontcheva, K. and Cunningham, H. (2003), Robust generic and query-based summarization, *10th Conference of the European Chapter of the Association for Computational Linguistics, EACL-2003*, Hungary.

Voorhees, E. M. (2003), Overview of the trec 2003 question answering track, *Proceedings of the Twelfth Text REtrieval Conference (TREC 2003)*.

Voorhees, E. M. and Tice, D. M. (2000), Building a question answering test collection, *SIGIR '00: Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, ACM Press, Athens, Greece, pp. 200–207.

Automatic Summarization of Meeting Data: A Feasibility Study

Anne Hendrik Buist, Wessel Kraaij and Stephan Raaijmakers

TNO TPD

Abstract

The disclosure of audio-visual meeting recordings is a new challenging domain studied by several large scale research projects in Europe and the US. Automatic meeting summarization is one of the functionalities studied. In this paper we report the results of a feasibility study on a subtask, namely the summarization of meeting transcripts. A Maximum Entropy based extractive summarization system using a mix of 15 features improved the performance of a baseline system selecting all utterances longer than 10 words with 20% (F-measure). However, stronger contextual awareness seems to be necessary in order to reduce the precision of the summarizer. The study required the creation of reference extractive summaries, which is documented in the paper.

1 Introduction

As speech recognition of broadcast news is becoming more mature, research is moving into types of speech that are more challenging. One such area is conversational speech. Initially telephone conversations were studied but more recently attention moved to meeting recordings. Indeed, interesting applications can be foreseen if automatic speech recognition (ASR) performance of conversational speech could be boosted to reach the same level of accuracy as for broadcast news. In the EU projects M4 (M4 2002b) and AMI (M4 2002a), meetings are recorded in a "smart meeting room" using multiple synchronized cameras and microphones (de Jong 2004). The key application developed in these projects is the "meeting browser", which facilitates users to search and browse meeting recordings. For this purpose the raw data is processed by multimodal analyzers that recognize "meeting actions" (e.g. discussion, presentation etc.) and perform a shallow semantic analysis. During the AMI project, topic segmentation and summarizing functions will be developed.

This paper describes a feasibility study of using machine learning techniques for extractive summarization of meetings. It is clear, that summarization is important for a meeting recording archive, since it will help users to find relevant meetings and (if the summary is linked to the recordings) to locate salient fragments of recordings for viewing. Reading summaries is much more time-efficient than listening to (or viewing) a recording. Also reading/searching the raw transcripts is not desirable, as these contain a lot of backchannels, elaborations and side topics which do not contribute to the content.

The area of summarizing dialogues or meetings has not yet been explored by many researchers. Some groundbreaking work has been done by Klaus Zechner and Alex Waibel, who created a dialogue summarizer DiaSumm (Zechner and Waibel 2000), which features include turn-linking, topic segmentation and information condensation. In (Zechner 2002), Klaus Zechner gives an overview of work done in this area. While one may think that the process of summarizing meetings is similar to the sum-

marization of news articles or broadcast news, it appears to be very different in practice. In contrast to written publications, sentence boundaries are very hard to discern, because conversations contain many disfluencies. Additionally, the information density in meetings is much lower than in news, which in essence is highly condensed information.

Moreover, trivial automatic summarization of news items is often facilitated by the fact that news articles have titles and lead text, and taking just the few initial lines of a news item already yields a good summary. For meeting summarization, such metadata is usually not available.

Our goal is to create a system which can extract all important topics from a recorded meeting and present them in an understandable way, thus creating a readable summary of the meeting. The summary should then be linked to the audio in a multimedia player, which allows for browsing the summarized meeting both audibly and textually. For an initial feasibility study, we investigated whether machine learning techniques that had proved to be successful for the summarization of broadcast news, could be adapted for the meetings domain. A fully functional automatic meeting recording summarization system would be highly complex, since it combines a.o. high quality speech recognition, speaker segmentation, utterance segmentation, dialogue act interpretation, domain knowledge with summarization techniques, each of which components are not sufficiently mature yet. Therefore, we performed a limited study into the effectiveness of structural and lexical properties of utterances as features based on manual meeting transcripts.

The rest of this paper is organized as follows: section 2 describes our approach to summarization using Maximum Entropy models, section 3 describes the corpus and annotation procedure, sections 4 and 6 describe the features used for the ME models and the experiments. The paper is finished with our preliminary conclusions and ideas for future work.

2 Maximum Entropy based summarization

Even though automating abstractive summarization is the goal of summarization research, most practical systems are based on some form of extractive summarization. Extracted sentences can form a valid summary in itself or form a basis for further condensation operations. Furthermore, evaluation of extracted summaries can be automated, since it is essentially a classification task.

During the DUC 2001 and 2002 evaluation workshops, TNO developed a sentence extraction system for multi-document summarization in the news domain. The system was based on a hybrid system using a Naive Bayes classifier and statistical language models for modeling salience. Although the system exhibited good results (Kraaij, Spitters and van der Heijden 2001, Kraaij, Spitters and Hulth 2002), we wanted to explore the effectiveness of a Maximum Entropy (ME) classifier for the meeting summarization task, as ME is known to be robust against feature dependencies. Maximum Entropy has also been applied successfully for summarization in the broadcast news domain (Osborne 2002).

2.1 Maximum Entropy Modeling

The maximum entropy model framework can classify information that comes from many different sources. The data the model trains on can be described as a vector of features $\{f_1, \ldots, f_k\}$. One of those features could be as follows:

$$f_j(b,c) = \begin{cases} 1 & \text{if SegmentLength}(c) = Long \;\&\; b = \texttt{good} \\ 0 & \text{otherwise} \end{cases}$$

This feature teaches the model that a segment that is *long* (for instance, longer than 20 words), is relevant to the summary, and the probability $p(\texttt{good}, c)$ will increase. b can either be `good`, which means it should belong in the summary, or `bad`. These features can be of a complex type, and we can use prior knowledge about what information is important for classification. Each feature that is in the model corresponds to a certain constraint. Then from all the models that satisfy the constraints:

$$\sum p(b,c) f_j(b,c) = \sum \tilde{p}(b,c) f_j(b,c), 1 \leq j \leq k$$

the one that maximizes the entropy $H(p)$ is chosen:

$$H(p) = -\sum p(b,c) \log p(b,c)$$

$\tilde{p}(b,c)$ is the observed distribution of features found in the training data. Choosing this maximum entropy model is a method to preserve as much uncertainty as possible: we want to have as little unjustified constraints of information as possible (Manning and Schütze 1999, Ratnaparkhi 1996): when the model finds a segment to be `good` or `bad`, we know it has found sufficient evidence for this outcome.

This model has proved to be very useful for many natural language processing tasks, including sentence detection, named entity recognition and part of speech tagging. We have used the OpenNLP implementation of the model, which is freely available on the web (Baldridge, Morton and Bierner 2001).

2.2 Applying ME for meeting summarization

As for any application of supervised machine learning techniques, a corpus is required that is annotated with ground truth information. For our experiments, the ICSI Meeting Recorder corpus (we will call this ICSI corpus from now on) was used, available from LDC. The M4 project did not have an extensive corpus available, especially not with manual transcriptions of the audio. At the time of the experiments, no ground truth extractive summarization data was available, so we manually annotated several meetings from the ICSI corpus. The annotation procedure is described in section 3. Subsequently several lexical and structural features were selected (some features that have been successfully applied in the broadcast news domain were evaluated as well). A subset of the annotated data was used for training the ME classifier. Training itself was based on determining (feature=value) pairs for all features for each sentence. The feature selection process is described in more detail in section 5.

3 Annotation procedure

A significant amount of time of the feasibility study was spent on producing training material for the automatic summarizer. Unfortunately, we did not have time nor the people let multiple people annotate the same meetings. We chose to have one annotator instead, who did his best to annotate 6 meetings, approximately an hour each. By doing the annotation in multiple steps, backtracking and correcting, we tried to guarantee that the quality of each summary was decent. Annotating the meetings took about 12 to 14 hours per meeting[1]. Approximately 22000 segments were rated in this way.

3.1 The ICSI Meeting Recorder corpus

The meeting corpus developed at ICSI, Berkeley (USA) consists of about 75 meetings recorded at ICSI. The meetings of several research groups at ICSI were recorded, so conversations have a highly technical focus. For each meeting, transcripts are available that contain both the start and end times, of words and speaker segments. In addition, the original audio recordings are available and several hand-annotated analyses of the corpus, e.g. of dialogue acts (provided in the MRDA corpus) and adjacency pairs[2].

3.2 Extract-based summaries

All segments of the six ICSI meetings were annotated for importance on a ternary scale. A segment in this context is a whole sentence or a part of it, spoken by one person. Sentences in the corpus were automatically cut off at some points, where the speaker would pause for a certain amount of time. Segments rated with 3 are highly relevant for the summary, while a 1 indicated that they are of little or no importance. Rating with a 2 indicates either an (ongoing) elaboration on the subject, or expresses doubt by the annotator regarding the importance of the segment.

3.3 Annotation method

In order to keep structure in the annotation, a few rules were followed while annotating. First of all, the annotator attempted to base summary annotations on just the text and specifically avoided to be biased by his knowledge of NLP techniques and the problems that specific utterances would pose. We chose to annotate the meetings in the MRT format.[3] During the actual annotation the following scheme was followed (for each meeting):

- Part I

[1] Meetings are often unstructured, and the audio can be very hard to perceive

[2] In conversations, many utterances are directed to evoke a natural response. E.g., complaints require apologies or maybe counter-complaints. This is called an adjacency pair.

[3] This is the XML format used by the ICSI corpus. The Meeting Recorder Dialogue Act (MRDA) corpus uses a different format, but that is deprecated.

1. Make a printout of all segments, preceded by the speaker of the utterance. Every segment on a new line.
2. Scan the printout to detect the topic.
3. Sequentially read the printout, rating segments on the fly. Backtrack when encountering a possible flaw (which could be e.g. incoherence, or things which seemed unimportant at first).
4. Listen to the audio when in doubt.
5. Import the handwritten segment ratings into the MRT file using a simple PERL script. While doing this, additional flaws in the summary were sometimes detected and in that case corrected.
6. Correct any mistakes made in importing by manually editing the MRT file.

- Part II

7. Make a printout of the important segments (rated 3). Calculate the percentage of the summarized portion over the whole meeting.
8. Recheck for fluency, understandability and other errors encountered. These were manually corrected again in the MRT file. When the percentage of extracted text (in words) was more than 30%, the summary was also more thoroughly checked for superfluous portions, and normalized. All corrections were also annotated on the original printout.

Accidently we used an older version of the ICSI corpus to annotate, so we had to re-rate all the meetings for the new format, because sentence boundaries had changed between versions (Sentences were cut off at different points). In this process, all segments were checked again quickly and some more corrected.

3.4 Evaluating importance

A crucial part in summarization is how to judge whether a segment is important or not. Other than in news articles or papers, things are said more than once in a meeting. It is hard to decide, when two utterances are almost equal, which one should make it into the summary. It is not good to include both, because the resulting summary would contain redundant parts. When evaluating, the automatic summarizer may prefer one of those two sentences above the other, for some reason, which might be the 'wrong' one. We found no solution for this, except that the annotator chose the most logical one in this case. For example, when the two sentences would be uttered by two different participants, one restating the former, most logical would be the original expression. In other, clearer cases where the segments differed a little, the more elaborate one was chosen, observing that the extra information was relevant.

When a participant starts a long series of utterances, selections were made on very clear points only. Even though a sentence might accidentally be understandable when removing a portion, it is not a wise thing to do.

Sometimes, when a segment consisted of a conjunction only, such as 'And um', these segments were also rated as good, because it would interrupt the flow of words

in the summary, when the segments are 'stitched' back together lateron. An exception is when such a segment contains only a backchannel: removing this does not harm the flow of a sentence so can be safely removed, resulting in a score of 1.

In some cases, a segment is only partly interesting. The whole segment was marked as important (3). This is one of the reasons that an abstract-based summary will probably be able to include the same information in a more condensed form.

What is important regarding the content may also differ on a per audience basis. The annotator assumed the summary to be a recollection of topics discussed in the meeting, trying to preserve as much information as possible.

3.5 Result of corpus summarization

The annotation eventually resulted in a rated corpus of 6 meetings, which incorporates approximately 6 hours of dialogue. The compression-rate in words is about 70%, and 80%-90% on the segment level. This difference is due to the fact that very long sentences are of most importance in a summary, and ultra-short ones are often non-salient. In addition to rating every segment, a topic segmentation was also performed: every meeting was annotated with the topics that are brought up during that meeting. This data was not (yet) actually used in the summarizer.

4 Experiments

4.1 Training and testing

The six hand-annotated meetings were divided into a training (4 meetings) and evaluation set (2 meetings). Training involved two steps: a feature extractor computed feature vectors with key=value pairs for each segment in the training set. Subsequently, the feature vectors (complemented with the truth data) were used as input to train a maximum entropy model. This model was applied to predict the salience value of segments of the test meetings. By counting how many segments in the testdata were correctly labeled `good`, the performance level of the model could be quantified. This performance was measured by the recall (how many of the total relevant segments are recognized correctly) and precision (percentage of relevant segments in relation to the number of segments labeled as `good`). In addition their harmonic mean (the F-measure) was computed (Van Rijsbergen 1979):

$$F - Measure = \frac{2 * Recall * Precision}{Recall + Precision}$$

Additionally, 10-fold cross-validation was used because of the relatively small dataset, and to create more stable results. The maximum entropy model tends to perform better when trained on a balanced set of good and bad examples, so the good examples in the training data were randomly oversampled. This process gets random good samples from the training set and adds them at the end. A 40% oversample rate makes sure that the good samples cover 40% of the training data. A 50% rate means a perfect balance between good and bad samples. The initial balance between good and bad

samples is approximately 17% good, 83% bad. Shown samplerates always indicate the percentage of good samples.

5 Feature generation and overview

In order to generate a good model, it is important to find as many strong features as possible. A feature can be something like frequent words or sentence length, or can for instance capture a relation to previous segments. Although we found many features that weighted towards being summary-specific, no strong features could be found. This was actually a bit of a surprise, because we initially expected that recycling the feature set of the TNO summarizing system that was developed for DUC2002 would generate a solid base. However, by finding many mediocre features it is still possible to achieve acceptable results. Table 1 provides a table with all the used features, a short explanation and the abbreviation that is used in the results.

SL	Sentence length: a segment is either ultra-short, short, medium or long.
MIN	The segment contains more than 3 words.
LW	Segment contains long words (more than 10 characters).
TF	Segment contains words that stood out in a TF-IDF approach.
FR	Segment has frequent words (own algorithm, many features).
FR1	Segment has frequent words (own algorithm, one feature).
FRB	Segment has frequent bigrams.
ACK	Segment is an acknowledgment.
CUE	Segment contains cue phrases.
CON	Segment connects with previous sentence (= same speaker and continues the sentence)
EMP	Segment is empty (no words).
SAM	Segment has the same speaker as previous one.
IMP	Segment features important speakers (3 features of the most prominent speakers).
DA	A number of features for some relevant dialogue acts (using the MRDA corpus).
DIG	Digit task.
TUR	A turn change took place: the next speaker is different from the last.
LON	A long turn: a speaker takes a turn over many segments, uninterrupted.
PC	Whether the previous segment is good. 2 features, go back 2 segments.

Table 1: Feature overview

Some features are actually a set of features which belong together: for instance, the SL feature set counts four features, of which only one can be true at the same time. This approach is needed for correct implementation by the Maximum Entropy model. The purpose and usage of some features may speak for themselves, but a few need additional explanation.

5.1 Sentence Length (SL)

This feature class is divided into four parts: ultra-short, which are segments that are only 10 characters long or less. There are many occurances of ultra-short segments which are often of no importance. The boundaries for 'short' are larger than 10 and smaller than 30 characters. 'Medium' is between 30 and 80 characters, and everything longer than 80 is considered 'long'. The measure is in characters instead of words especially for smaller segments: a segment like 'I like it' contains no information, while a sentence with three long words tends to be more important.

5.2 TF-IDF (TF)

This is an implementation of the TF-IDF information retrieval technique (Baeza-Yates and Ribeiro-Neto 1999) where words are highlighted that occur more often in the current meeting than the average in a corpus of similar meetings. Segments that contain document-specific words often contain valuable information for a summary. The feature this generates indicates whether such a word is in a segment. Only the top 20 words are selected.

5.3 Frequent words and bigrams (FR FR1 FRB)

A variation on the TF-IDF algorithm, this implementation uses the mean and standard deviation of word frequencies in a corpus to evaluate their importance. When tested as a single feature, it works slightly better than TF-IDF. The variant implements a feature for the 15 most important words to occur in the document, which results in 15 different features. This method gives the model many more features to train with. The bigram version implements the same for significantly important sets of two words. To ensure a clean list of important words in the unigram variants, a closed word stoplist is used: the use of particular closed words are not a sign of importance, they only depict the style of the speakers.

5.4 Cue phrases

Although this feature is not fully implemented, it captures the idea: people tend to use certain phrases to announce something important. To find some phrases that stood out from the good segments, we experimented with likelihood ratio statistics, which can distinguish specific terms (or phrases) by comparing occurrence frequency with a background corpus. When using bad segments as the background corpus, one would be able to see which phrases are specific for a summary. Unfortunately, this did not work as well as we hoped: the results were words/phrases that were not obviously important. We handpicked a number of cue phrases from the top list. When those occur in a segment, this feature is triggered.

5.5 Linking to the previous segment (PC)

This can be done by doing a run on the generated list of feature vectors, and add new features that determine if the previous segment was rated good. This results in two features, one for the last, and one for the second-last segment. It adds a new layer of uncertainty, because the features are based on the assumption that the segments were correctly labeled in the first run.

5.6 Important speakers (IMP)

By comparing speaker times in the rated (test)corpus, we found that the most frequent speakers also say more important things. This behavior is captured in three features that name the first, second and third longest speakers. This is calculated by summing up the time of speech for each individual speaker.

5.7 Dialogue Acts (DA)

A dialogue act is metadata about a segment that informs about the intention of the speaker. Examples are grabbing the floor[4], making a statement, or asking a question. Every segment can contain multiple dialogue acts. When experimenting with them, we found that interesting segments were mostly statement-only and question segments.

We had the privilege to have access to the MRDA corpus, a set of meetings from the ICSI Meeting Recorder project where dialogue acts were hand-annotated. Unfortunately, because the format of the ICSI corpus was still changing, the MRDA corpus was not completely compatible with the newer ICSI format: segments were wrapped at irregular places, which prevented a clean remapping of the corpus. The eventual mapping is probably accurate for at least 90% of the segments, which makes it quite usable. Eventually an automatic DA tagger will be implemented.

6 Results

The system was trained with different feature combinations, because using the complete feature set rendered suboptimal results. A result was calculated for every single feature, after which the best result was selected. Then every feature combined with that best feature was evaluated again. This was done a few times. In table 2 the initial result per feature is shown. A number of features have no significant results because the model selected all segments to be important. The strength of these features is their combination with others.

We also experimented with oversampling percentages. Unfortunately, due to using random samples for oversampling, results differed between identical runs of the program: the deviance was approximately 1%. As expected, oversampling greatly improves the summarizing process, as can be seen in table 3. Because at 50% the last

[4] *Grabbing the floor* is interrupting the current speaker, in order to make a statement. Cp. *Holding the floor*, in which the speaker tries to keep the attention by connection his statements, while he prepares his words (e.g. with ‘and umm…’)

Feature	TF	FR1	FR	FRB	CUE	LW	ACK	DA	MIN
F-Measure	0.2	0.321	0.31	0.233	0.157	0.392	0.144	0.314	0.269

Table 2: Single feature results

Oversampling	-	20%	30%	40%	50%	60%	70%	80%
F-Measure	0.162	0.283	0.406	0.447	0.503	0.508	0.507	0.496

Table 3: Results for oversampling, the percentage shows the amount of good samples in the set. This is 17% without oversampling.

Feature set	*Recall*	*Precision*	*F-Measure*
Baseline (random)	0.199	0.169	0.182
Baseline (10 words)	0.448	0.362	0.401
All features on	0.641	0.379	0.476
All features - PC	0.653	0.381	0.482
TF FR1 FRB CUE (content features only)	0.228	0.348	0.276
LW SL LON ACK CON EMP SAM IMP DIG TUR DA MIN PC (no contentfeatures)	0.620	0.354	0.451
LW SL LON TF FR ACK CUE CON EMP SAM IMP DIG TUR (optimal feature set)	0.682	0.401	0.505

Table 4: Results with different feature sets, 50% oversampling

great increase was noted, this rate was used in further tests. With the increase of the samplerate, a sharp increase of recall is seen, extracting more segments than wanted from 50% onward.

For comparison, we also tested two baseline approaches. The first is a random extract of 10% of the segments from each test meeting. This is the absolute baseline, as there is no coherence or logic in this method. The second baseline selects those segments that contain more that 10 words. This is motivated by the fact that there are many backchannels and filtering these out is a simple first step to clean up a transcript for extractive summarization.

System	Our system	N-Top	Fung Ngai Chi-Shun	MEAD
Result	50%	41%	69%	59%

Table 5: Results for some document summarization systems. N-Top is a baseline that extract the n top sentences from each document, Fung and MEAD are both multiple document extraction based summarizers. Percentages are the amount of correctly extracted sentences. Single document summarizers achieve even higher percentages.

6.1 Inter annotator agreement

The optimal selection of features renders a F-Measure of 0.505, which is not a very high score. This relatively low performance may have a good reason, that cannot be easily solved: a paper by Mandar Mitra et al. (Mitra, Singhal and Buckley 1997) addresses the issue of human agreement. They let two people make a summary of the same text, where the persons had to choose critical paragraphs that were to be included. The overlap between these persons was only 46%, which means that they agreed only on 46% of the content of the final summary. This is a serious problem, because this means that only half of a summary's content is typical. While manually creating summaries, for every segment a choice has to be made, which in some cases inevitably leads to arbitrariness. For a machine learning model to be effective, it is necessary that the data is somehow consistent. Because we only had summaries made by one individual, we had no possibility of comparison.

6.2 Other issues

Unfortunately, there are no similar systems available that deal with meeting summarization to compare the results with. When compared to summarization of broadcast news or documents, performance is quite low (See table 5 for some results taken from (Fung, Ngai and Cheung 2003)). However, these are completely different types of content. One of the big differences is the nature of a meeting compared to news articles and other written documents: where articles always follow strict guidelines for publishing, meetings do not. A meeting has little structure in topic sentences or any placement of important segments. Where the first sentence of an article often contains the topic, this is rarely the case with meetings. Sometimes a meeting will start right away, other times there will be some chit-chat beforehand. It is very hard to discern between such conversations.

6.3 Screen output in SMIL

When a summary has been generated, it is possible to listen to an audio version using Realplayer. Realplayer supports SMIL (Synchronized Multimedia Integration Language), which is a XML markup language for audiovisual presentations. A SMIL version of the extract was produced, consisting of just the extracted segments in textual form (they were displayed as running text), with synchronous presentation of the

corresponding audio, thus skipping the material marked as unimportant. Segments are colorcoded for each individual, and the format also allows clicking in a topic index to skip certain parts. This system actually works very well and feels quite natural, even though segments are sometimes not completed. Because people involved in conversations often do not finish their sentence either, this does not lead to irritation.

7 Conclusion

The automatic sentence extraction system was able to improve a heuristic baseline system by about 20%, showing the effectiveness of the chosen features. Nevertheless, the absolute performance of the extractor is less than is expected for NLP classification problems. We feel that lower levels of performance (in terms of F-value) are not so surprising for summarization tasks, since it is well known that human annotators have a low level of agreement on a manual task. Overall, the system produces fairly readable summaries, even though no effort has been done to rephrase sentences. The bottleneck of the system is the lack of structure in meetings, and related to this the absence of good features.

Furthermore, the study gave some insight into the structure of meetings, showing some interesting features that could be used in further research. The approach to classify each segment individually, without looking at the context is obviously too naive. Still, our results can function as a reference baseline for comparison with future results.

8 Future work

To continue work on this matter, a new approach using lexical chains is investigated. Lexical chains are capable of pinning down topic hotspots in a document, and connecting the most important sentences. The use of lexical chaining can be implemented as a whole new method, or as a enhancement on the feature set of our current summarization system, e.g. by producing better (context based) estimates of which tokens are topical.

References

Baeza-Yates, R. A. and Ribeiro-Neto, B. A.(1999), *Modern Information Retrieval*, ACM Press / Addison-Wesley.

Baldridge, J., Morton, T. and Bierner, G.(2001), The maximum entropy framework, http://maxent.sourceforge.net/about.html.

de Jong, F.(2004), Disclosure of non-scripted video content: InDiCo and M4/AMI, *Proceedings of CIVR 2004*.

Fung, P., Ngai, G. and Cheung, C.-S.(2003), Combining optimal clustering and hidden markov models for extractive summarization, *Proceedings of the ACL 2003 Workshop on Multilingual Summarization and Question Answering*.

Kraaij, W., Spitters, M. and Hulth, A.(2002), Headline extraction based on a combination of uni- and multidocument summarization techniques, *Proceedings*

of the ACL workshop on Automatic Summarization, Document Understanding Conference (DUC 2002), Philadelphia, USA.

Kraaij, W., Spitters, M. and van der Heijden, M.(2001), Combining a mixture language model and naive bayes for multi-document summarization, *Proceedings of the Document Understanding Conference*, Document Understanding Conference (DUC 2001), New Orleans, USA.

M4(2002a), Augmented multiparty interaction (ami), http://www.m4project.org/overview.html.

M4(2002b), Multi modal meeting manager (m4), IST-2001-34485 http://www.m4project.org/overview.html.

Manning, C. D. and Schütze, H.(1999), *Foundations of Statistical Natural Language Processing*, MA: MIT Press, Cambridge.

Mitra, M., Singhal, A. and Buckley, C.(1997), Automatic text summarization by paragraph extraction, *Mani and Maybury*, MIT Press, Cambridge, Massachusetts.

Osborne, M.(2002), Using maximum entropy for sentence extraction, *ACL 2002 Workshop on Automatic Summarization*.

Ratnaparkhi, A.(1996), A maximum entropy model for part-of-speech tagging, *in* E. Brill and K. Church (eds), *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, Somerset, New Jersey, pp. 133–142.

Van Rijsbergen, C. J.(1979), *Information Retrieval, 2nd edition*, Dept. of Computer Science, University of Glasgow.

Zechner, K.(2002), Summarization of spoken language - challenges, methods and prospects.

Zechner, K. and Waibel, A.(2000), Diasumm: Flexible summarization of spontaneous dialogues in unrestricted domains, *Proceedings of COLING*, International Conference on Computational Linguistics (COLING), Saarbrucken, Germany.

Phases and Complexity in Phrase Structure Building

Cristiano Chesi

University of Siena - MIT

Abstract

The *Minimalist Program* (Chomsky 1995–2001) sketches a model that aims to be empirically adequate and theoretically motivated but that does not fit in any clear way with specific performance algorithms such as *parsing* or *generation* even though much emphasis is put on "interface properties". In this paper I propose that a *Minimalist Grammar* formalization (an extension of Stabler's 1997 proposal) can be used both in *parsing* and in *generation* if we re-orient the directionality of the *Structure Building Operations* (*merge* and *move*) and if we formalize the notion of *phase* (Chomsky 1999). This will help us to define generalized algorithms, suitable both for *parsing* and for *generation*, which are computationally tractable in dealing with *ambiguities* and *long distance dependencies* resolution.

1 Introduction to minimal grammar specification

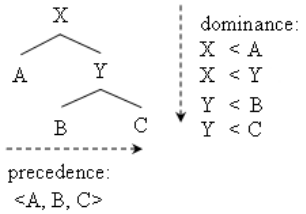
Formalizing a linguistic theory forces us to specify everything we need to describe a language. When choosing a specific formalization we must consider to which extent it encodes linguistic intuitions and at which computational cost it does. In this work I will deal essentially with this second issue,[1] providing some formal argument in favor of the necessity of limiting unbounded long distance *Structure Building Operations* (such as *move*, Chomsky 1995–2001) by means of a precise formalization of the notion of *phase* (Chomsky 1999). In order to provide the minimal background for this discussion, we need to define explicitly the notion of *Structural Description* (§1.1), a precise formalization of *Minimalist Grammar* (§1.2) and a definition of *parsing* and *generation* (§1.3). This should allow us to understand the necessity to (re)define some essential property of the *Structure Building Operations* (*merge*, *move* and the idea of *derivation by phase*, (§1.4)) because of empirical and computational considerations.

1.1 Structural Descriptions in terms of immediate relations

It is a standard assumption to consider a sentence as a bidimensional entity bearing information on both *precedence* and *dominance* relations among lexical items, where *precedence* represents a total order among pronounced elements (namely words, that are groups of phonetic features) while *dominance* expresses the constituency/dependency relations among pronounced and other implied (abstract) elements (semantic and other syntactic features like phrase identifiers). These two kinds of information can be encoded within tree-like structures such as the following one:

[1] See Chesi (2004) for a discussion of the empirical issue.

(1)



From a formal point of view, a *Structural Description* (SD) can be expressed as follows:

(2) $\mathbf{SD}_{\text{standard}}$ = {**I**, **P**, **D**, **V**, **A**} such that

I is a *precedence* order (a total strict order, that is a binary, transitive and asymmetric relation, defined on any element belonging to the subset I_T of I such that I_T is the set of terminal elements; e.g. {the, dog, is, black})

D is a set of *dominance* relations (a partial strict order, that is a binary, transitive and asymmetric relation, defined on some elements of I; e.g. "D" dominates "the", "N" dominates "dog", "DP" dominates "dog" etc.)

V is a finite set of *vertices* (the nodes in the tree)

A is an *assignment function* from V to I

I is a finite set of *identifiers* (e.g. {the, dog, is, black, DP, D′, D^o, N^o ...})

Leaving aside both **V** and **A** (maybe superfluous as discussed in Chesi 2004) I wish to redefine the other three elements in such a way that *precedence* and *dominance* only hold between immediately adjacent objects:

(3) $\mathbf{SD}_{\text{revisited}}$ = {**I**, **P**, **D**} such that

I is a finite set of *identifiers* (minimally, if we accept the inclusiveness condition, Chomsky 1995, we should consider nothing but lexical items and their features)

P is a finite set of *immediate precedence* relations (different from the classic total strict order, this relation is only defined between two adjacent items)
for example ("<A, B>" means "A *immediately precedes* B"):
SD = [$_A$ A [$_B$ B C]] → P = {<A, B>, < B, C>}

D is a finite set of *immediate dominance* relations (different from the classic *dominance* relation, this one is a partial, binary, intransitive and asymmetric relation
for example ("A < B", means "A *immediately dominates* B"):
SD = [$_A$ A [$_B$ B C]] → D = {A < B, B < C}

These restricted versions of *precedence* and *dominance*, defined only between adjacent elements, encodes in a very transparent way significant linguistic relations such as *merge* (Chomsky 1999): "A < B", in fact, corresponds to "A *merges* with B and *projects over* B" (namely A is the *head* of the constituent resulting from *merge*):

(4) $A < B = \text{merge}(A, B) \rightarrow [_A\ A\ B]$

On the other hand, the necessity to describe phrase structures also in terms of discontinuous constituency dependencies (chains/movements, binding relations, ellipses), can be captured within the revisited SD formalization given in (3), following the intuition that an element is "merged" in more than one position within the phrase structure, even though it is (fully) pronounced (then *phonetically linearized*) only in one of these positions (usually the structurally highest one). The interplay between *immediate dominance* and *immediate precedence* defined among lexical elements in the phrase structure can help us to capture this intuition:

(5) **Long Distance Dependencies** (definition)

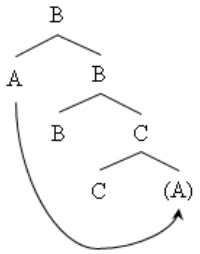
two non-empty elements enter a *long distance dependency* (thus forming a discontinuous constituency relation) when an *immediate dominance* relation but no *immediate precedence* relation is defined between them.

Note that a phonetically null element is not necessarily an empty element (since semantic and other formal features should be present). For instance given the following SD, A and C enter a Long Distance Dependency since C<A exists but neither <A,C> nor <C,A> is present:

(6) **I** = A, B, C
P = <A, B>, <B, C>
D = B < A, B < C, C < A

This SD can be graphically represented by the tree below:

(7)



The Long Distance Dependency reported in (6)–(7) is essentially an instance of *movement* (Chomsky 1995-2001) even though the directionality of the arrow is inverted to indicate that the highest element provides feature values for interpreting the underspecified features on the lowest "copy").

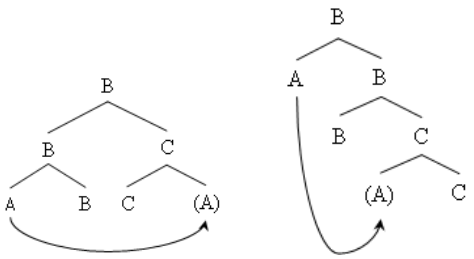
There are empirical reasons to believe that it would be possible to extend this approach to *pronominal binding* (on the line of Kayne 2002) and to *control* (Hornstein 1999, Boeckx, Hornstein 2004) capturing major typologies of Long Distance Dependency, even though I will not discuss the issue in these pages.[2] Note that a SD defined

[2] Again, refer to Chesi (2004) for a full discussion of the empirical scope of the proposal.

as in (3) with simply the information given in (6) is not able to discriminate among the right branching structure given in (7) and other structures such as the ones drawn below:

(8)

a. b.



Without using *assignment functions*, as proposed in (2)), we can rule out the unwanted representations by posing extra constraints on the occurrence of *long distance dependencies* and on the general shape of the SD. The *Linear Correspondence Axiom, (LCA)* (Kayne 1994), for instance, would suggest a deterministic mapping between *dominance* and *precedence* depending on the structural function of the lexical items involved in the SD strictly predicting a right branching structure. With the same spirit, I will adopt the following principle:

(9) **Linearization Principle** (inspired to *LCA*, Kayne 1994)
If A $<$ B, then either
a. $<$A, B$>$ if B is a *complement* of A (that is, A selects B), or
b. $<$B, A$>$ if B is a *functional projection*[3] of A

Following the notion of *merge* (in terms of *immediate dominance*) given in (3), we can define an useful asymmetric relation among the nodes within a tree:

(10) **Asymmetric C-Command** (definition)
When two elements *merge*, they respectively *asymmetrically C-command* all constituents of their sister

This definition is sufficient to discard (8.a) under the relatively standard constraint on movement provided below:

(11) **Constraint on movement** (definition)
A moved element always *asymmetrically C-Commands* its trace(s)

As it will be clear later on, we do not need to discard (8.b) since within this formalization the *linearization* of a trace is irrelevant.[4] Then (8.b) and (7) are equivalent.

[3] Assume *functional projection* to be synonym both of *specifier* and of *functional position*, following Starke (2002).
[4] But see Nunes (2004) for a different perspective.

1.2 Minimalist Grammars

We can think of a language, that is an infinite set of grammatical expressions each associated at least with a single grammatical SD, as a way of productively restricting the theoretically possible *precedence*/*dominance* relations that can co-occur within the same sentence. We refer to the intensional procedure that characterizes this set, as speaker's *competence*. Formally speaking, this *competence* can be described by a *grammar* that, following the minimalist trend, includes, at least, the specification of a *lexicon* (a finite set of *words* built from an *alphabet* with associated specific *features*) and a set of *Structure Building Operations*.

Stabler (1997) proposes a clean formalization of a *Minimalist Grammar* (*MG*) as outlined in Chomsky 1995 that includes these two basic components (*Lexicon* and *Structure Building Operations*). Following his proposal, a MG can be defined as a 4-tuple {V, Cat, Lex, F} such that:

(12) **Minimalist Grammar** (**MG**, Stabler 1997)

V is a finite set of non-syntactic features, $(\pi \cup \sigma)$ where π are phonetic features and σ are semantic ones;

Cat is a finite set of syntactic features, Cat = (*base* ∪ *select* ∪ *licensors* ∪ *licensees*) where

base are standard categories {complementizer, tense, verb, noun ...},

select specify one of the three possible kinds of selection {=x, =X, X= | x ∈ *base*} where =x means simple selection of an x phrase, =X selects an X phrase, suffixing the selecting head with the phonetic features of the selected X phrase; X= selects an X phrase, prefixing the selecting head with the phonetic features of the selected X phrase,

licensees specify requirements forcing phrasal movement {-wh, -case ...}, -x triggers covert movement, while -X triggers overt movement,

licensors are features that can satisfy licensee requirements {+wh, +case ...};

Lex is a finite set of expressions built from **V** and **Cat** (the *lexicon*);

F is a set of the two partial functions from tuples of expressions to expressions {*merge*, *move*};

The language defined by such a grammar is the closure of the *lexicon* (*Lex*) under the *structure building operations* (*F*). (13) is a simple example of MG able to deal with simple *wh*- movements[5]:

[5]For the sake of simplicity, let us assume that capital features directly *select* the position of the arguments without involving pre/in-fixing (then, =X means that the argument X is *selected* to the right of the selecting head, while X= to the left). The very same result is however derivable by a combination of standard (non directional) *selection* plus a trigger for *movement* (for instance -case) or assuming, as Stabler does, a difference in the merge result whenever *complex* or *simple* trees are merged.

(13) MG example

V = π = {/what/, /did/, /you/, /see/}, σ = {[what], [did], [you], [see]}

Cat = *base* = {D, N, V, T, C}, *select* = {=D, =N, =V, =T, =C}, *licensors* = {+wh}, *licensees* = {-wh}

Lex = [$_{-\mathrm{wh}\ \mathrm{D}}$ what], [$_{=\mathrm{V}\ \mathrm{T}}$ did], [$_{\mathrm{D}}$ you], [$_{=\mathrm{D}\ \mathrm{D}=\ \mathrm{V}}$ see], [$_{=\mathrm{T}\ +\mathrm{wh}\ \mathrm{C}}$]

F = *merge*, *move* such that:

merge (X, Y) = is a function taking two adjacent subtrees X and Y, outputting an unified structure Z of the form [$_{\mathrm{X}}$ X Y] if and only if X has as a first selecting feature (=f, =F, F=) and Y has the needed selected feature F as the first feature of the base set

move (X, Y) = if a function taking two subtrees [$_{+\mathrm{g}}$ X] and [$_{-\mathrm{g}}$ Y] such that <[$_{+\mathrm{g}}$ X], W, [$_{-\mathrm{g}}$ Y]> (where W can be any possible subtree, even null, but without any selecting/selector feature *g* in it) and produces Z of the form [[$_{\mathrm{X}}$ Y X] W, t_{Y}]

Following Chomsky, a derivation proceeds *from-bottom-to-top*[6] and *licensees* trigger *movement* as shown below[7]:

(14) **1.** *merge* ([$_{\mathbf{=D}\ \mathrm{D}=\ \mathrm{V}}$ see], [$_{-\mathrm{wh}}$ **D** what]) → [$_{\mathrm{see}\ \mathrm{D}=\ \mathrm{V}}$ see, $_{-\mathrm{wh}}$ what]

2. *merge* ([$_{\mathbf{D}}$ you], [$_{\mathbf{D}=\ \mathrm{V}}$ see, $_{-\mathrm{wh}}$ what]) →
[$_{\mathrm{see}}$ you, [$_{\mathrm{see}\ \mathrm{V}}$ see,$_{-\mathrm{wh}}$ what]]

3. *merge* ([$_{\mathbf{=V}\ \mathrm{T}}$ did], [$_{\mathrm{see}}$ you, [$_{\mathrm{see}}$ **V** see, $_{-\mathrm{wh}}$ what]]) →
([$_{\mathrm{did}\ \mathrm{T}}$ did, [$_{\mathrm{see}}$ you, [$_{\mathrm{see}}$ see, $_{-\mathrm{wh}}$ what]]]

4. *merge* ([$_{\mathbf{=T}\ +\mathrm{whC}}$], [$_{\mathrm{did}}$ **T** did, [$_{\mathrm{see}}$ you, [$_{\mathrm{see}}$ see, $_{-\mathrm{wh}}$ what]]]) →
([$_{\mathrm{C}\ +\mathrm{wh}\ \mathrm{C}}$, [$_{\mathrm{did}}$ did, [$_{\mathrm{see}}$ you, [$_{\mathrm{see}}$ see, $_{-\mathrm{wh}}$ what]]]])

5. *move* ([$_{\mathrm{C}}$ **+wh** $_{\mathrm{C}}$, [$_{\mathrm{did}}$ did, [$_{\mathrm{see}}$ you, [$_{\mathrm{see}}$ see, **-wh** what]]]] →
[$_{\mathrm{C}}$ What, $_{\mathrm{C}}$, [$_{\mathrm{did}}$ did, [$_{\mathrm{see}}$ you, [$_{\mathrm{see}}$ see, t_{what}]]]]

Some interesting formal results show that there is a weakly equivalent *Multiple Context-Free Grammar* for any MG (then MG are included in the *Mildly Context-Sensitive* class of grammars, Michaelis 1998) and that a *recognizer* algorithm can be defined (both top-down and bottom-up) for MGs (Harkema 01). However, it is difficult to draw any computational/cognitive conclusion from these results, because either they are based on a *deductive parsing perspective* (Shieber and al. 1995) or on a *weak equivalence* between MGs and other formalisms (e.g. *Multiple Context-Free Grammars*): namely, these grammatical formalisms/derivations can produce the very same set of strings MGs will produce, but either they fail to associate (derivationally) the same structures to these strings or they encode *lexical items*, *features* and *structure*

[6]That is, from the inner verbal head, adding piecemeal *complements* then *functional specifications*, up to the most external heads.

[7]This is a very simplified version of derivation, to be taken only as example. It would be clearly possible including the subject movement too, but this would have been required extra steps in the derivation.

building operations in a less transparent way with respect to the linguistic intuitions that justified them. I believe that these two factors are indeed crucial parameters of evaluation, at least if the final goal of the formalization is that of making clear what the computational (and possibly psycholinguistic) implications are both in *parsing* and in *generation* if we want to use MGs effectively in these two contexts. A fundamental step is then to define precisely how the *parsing* and the *generation* tasks could be described in order to pose some effective "interface conditions" on the grammar formalism.

1.3 Two (sub-)problems for parsing and generation: ambiguity and long distance dependencies

Given a MG *G* defined as in §1.2 and assuming that a *Structural Description* (*SD*) can be expressed basically in terms of *immediate dominance* (*D*) and *immediate precedence* (*P*) as proposed in §1.1, we can define the *parsing* and the *generation* tasks (symmetrically) as follows:

(15) **The parsing problem** (definition)

given a finite set of *phonetic features* π (grouped by words) and a *precedence* total order among them, find the relevant set of lexical items *Lex*, compatible with π and the set of *(immediate) dominance* relations *D* among σ features associated to π in *Lex*, if possible, if not reject the input.

(16) **The generation problem** (definition)

given a finite set of *semantic features* σ and a finite set of *dominance* relations *D* among them, find the relevant set of lexical items *Lex* and the correct *linearization* among π features associated to σ in *Lex*, if possible, if not reject the input.

Both problems can be factored in two distinct sub-problems: a part of *lexical ambiguity resolution* (... "find the relevant set of lexical items Lex"... compatible with π/σ...) and a part of *structural mapping* (from *precedence* to *dominance* and vice versa). These problems are difficult to solve, because of a non-univocal mapping (*non-determinism*) between phonetic features and words (homophony/homography, polysemy etc.) and because of *discontinuous constituencies* (*long distance dependencies*) which, among other factors such as *PP attachment* and *constituents conjunction*, cause structural ambiguity. I assume that these two problems pose interesting constraints (much less abstract than usual "interface conditions", Chomsky 1995) on the grammar specification if we accept the intuitive idea that the very same *competence* (then the very same *grammar*) has to be used in both contexts. As it will be clear in the next paragraph, *Structure Building Operations* are especially affected by these considerations.

1.4 Structure Building Operations and directionality

There are empirical and formal reasons (Chomsky 1995-2001) to believe that a minimal specification of the grammar can include *Structure Building Operations* such as

merge and *move* (**F** in Stabler's MGs, §1.2). This seems to be plausible also from a performance perspective when we have to map *precedence* and *dominance* with respect to lexicalized feature structures both in *parsing* and in *generation* contexts according to the definition provided in §1.3. There are however theoretical and psycholinguistic reasons to believe that the *Bottom-to-Top* orientation of the derivation cannot be pursued neither from a *parsing* nor from a *generation* perspective: Phillips (1996), for instance, shows how *intermediate constituencies* built from *Top-to-Bottom/Left-to-Right* can account for important contradictory constituency tests (that is, when *movement* would predict a different constituency with respect to *scope relations* in sentence such as "John gave a candy to any children in the library on Sunday..."); on the other hand, the cyclic idea of movement (*move short*, Chomsky 1995) and the notion of *derivation by phase*, require purely formal *uninterpretable features* to trigger intermediate steps in any *long distance movement* context. This seems to be a non-deterministic, unsatisfactory solution and it is definitely not suitable from a *parsing* perspective.[8] Moreover psycholinguistic evidences show that both in *generation* (*false starts*) and in *parsing* (*garden paths*) the orientation of the processing could fairly be from-left-to-right/top-to-bottom; last but not least, as far as I know, no clear psycholinguistic evidence supports the bottom-to-top model.

Taking into account these cues[9], the definitions of SDs (§1.1) and MGs (§1.2), assume that any item is licensed within a SD if and only if it is *selected*[10] or it is a plausible *functional specification*[11] of a lexical head according to the *Linearization Principle*. Assume, furthermore, that *merge* could be defined as a binary function which takes two feature structures and unifies them (in the sense of *unification grammars*, Shieber 1986).

In order to account for the above mentioned problems, I will re-define *move* as a top-to-bottom/left-to-right oriented function which stores an *unselected* element in a sort of *memory buffer* and *re-merges* it at the point of the computation where the element is *selected* by a lexical head (according to the *select* features of this lexical head and to the *Linearization Principle*).[12] This modification would allow us to get rid of the *uninterpretable features* hypothesis, then removing the teleological behavior behind the movement conception as a feature driven, bottom-to-top operation.

Beyond the empirical adequacy of these redefinitions of the *Structure Building Operations merge* and *move*, which I will not evaluate in these pages (as explained in fn.1), it would be necessary to evaluate their computational impact. In fact, while the *merge* operations has a local scope and does not present dramatic complexity issues (at

[8] Neither the notion of *numeration* makes any clear sense in *parsing*, nor the idea of adding arbitrary *uninterpretable features* to the lexical items in order to "reconstruct" a movement operation.

[9] For more datails refer to Chesi (2004).

[10] From this perspective selection means both *C(ategorial)-selection* (then it operates on *Cat* features) and *S(emantic)-selection* (Pesetsky 1982), then it operates on *semantic* features).

[11] A sort of *licensor* specification in Stabler's terms. According to *Cartography* (Cinque 1999–2002) I assume an articulated *functional structure* above any lexical elements, moreover expecting that the legitimated position within the structure has to be evaluated in terms of relative scope of the items effectively present within the SD.

[12] The simplest possible device to store un-selected elements out of the SD would be a stack, First-In-First-Out memory; this simple proposal is able to account for argumental movement, topicalization and wh-movement. See Chesi (2004) for solutions with wider empirical coverage.

least once the feature set of the lexical items are clear) the long distance nature of the *move* operation seems to be allegedly onerous if not bounded in its potentiality: capturing arbitrarily distant relations would not be a welcome result neither empirically nor psycholinguistically; moreover it could be computationally very expensive.

Introducing the idea of *derivation by phase* seems to be clearly possible also in this Top-to-Bottom oriented framework and it could be the solution for many problems roughly introduced before: from this perspective, a *phase* can be thought as the minimal part of a Top-to-Bottom computational process in which all the *functional* and *selectional* specifications associated to a given lexical head (i.e. N(oun) or V(erb)) are satisfied. Crucially, a *phase* can be included within F as a "Top-to-Bottom expectation" (*phase projection*), as follows:

(17) **Phase Projection** (definition)

a projection of the minimal set of *D* relations, optionally underspecified for π or σ features, within the *SD* according to the *select* features present in the processed *lexical head*.

Note that in order to make this device computationally interesting we should distinguish between "active" and "inactive" phases: informally speaking, a phase is active or *open* when their elements can still enter new *merge* relations (either because the head of the phase has not been processed yet or because of *move* which stored some of these elements within the memory buffer); on the other hand, a phase become inactive or *closed* when any required relation has been established: once processed the (lexical) head of the phase, *Phase Projection* applies, according to the *select* features present in the phase head, and the phase is closed; any element still present in the memory buffer at this point is inherited (remerged) within the lower projected phases.[13]

Rephrasing these notions of phase in more formal terms, given a grammar $G = \{V, Cat, Lex, F\}$, an input i to be processed, composed by elements belonging to *Lex* (even if specified only for π or σ features), considers:

(18) **Phase** (definition)

a complete process (of *parsing* or *generation*) involving:

- a *Phase Projection* about a potential SD structure,
- a proper subset i_{p} of i such that any item in i_{p} is a lexical element (*N* or *V*), the *head of the phase*, or it is a *functional specification* of the head of the phase,
- a memory buffer *M* to store unselected items and retrieve selected ones, initialized as empty, unless some items are inherited from the non-empty memory buffer of a selecting previous phase (this inherited element will be part of i_{p} and will be legitimated within the phase by merging it at the relevant position of the left-periphery, edge of the phase, as phonologically null element, unless otherwise specified);

[13] In fact the whole story would be a bit more complex. For sake of simplicity this is however enough to understand the following discussion. See Bianchi, Chesi (2005) for a detailed proposal on the distinction between *nested* and *sequential phases*.

(19) **Phase Completeness** (definition)

a phase is complete (i.e. *closed*) iff the head of the phase is saturated, namely if any mandatory selectional requirement (direct object, indirect object etc.) is satisfied (a *Phase Projection* can be considered a complete phase in order to close the phase generating this top-to-bottom expectation);

(20) **Phase Complementation** (definition)

a phase takes only *complete phases* as complements;

(21) **Phase Selection Requirement** (definition)

a phase has to be *selected*; items in the memory buffer, at the end of the phase, can be transferred only to the memory buffer of the selected phase(s).

The goal of using *phases* is then to reduce the potential non-determinism of *ambiguities* and *long distance dependencies resolution* both restricting the space of the problem in an empirically adequate way and forcing the algorithm to bias its choices following precise Top-to-Bottom expectations (that is, a precise set of *immediate dominance* relations underspecified for some features).

2 Reducing Complexity using phases

The complexity of a problem is an expression of the resources (essentially *time* and *memory*) needed to solve the problem (Papadimitriou 1994). More precisely, it is a function of the size of the problem, determined at least by three factors:

(22) **a.** the length of the input (n);

b. the space of the problem (all states the system can attaint by correctly applying any legal rule);

c. the algorithm used to explore this space.

While a. is largely independent from the grammar, b. and c. are strictly determined by the linguistic theory. From a MG perspective, b. is mostly determined by the *lexicon* and by the *structure building operations merge* and *move*; in the *top-to-bottom* perspective just mentioned, c. is a procedure driven by the *Linearization Principle* that has to inspect items and recognize which one is *licensed* in a specific position and which one has to be *moved* and *re-merged* in another (lower) position. Let us explore the complexity of the two (sub)problems of *ambiguity* and *Long Distance Dependency* identification with respect to these three parameters.

2.1 The complexity of Ambiguity

Considering both lexical and semantic ambiguity the first (sub)problem can be stated as follows:

(23) **Ambiguity problem** (definition):

given an input *i*, composed by π (in *parsing*) or σ (in *generation*) features grouped by words, of length *n*, and a number *c* of possible *Part-of-Speech* (*PoS*)[14], assign to each word from the input at least one *PoS*, if possible, if not reject the input.

The *complexity* of the problem, considering a brute force algorithm to solve it, is at worse O(c^n) (assuming that any word could be ambiguous among all *PoS*). A more realistic complexity order can be guessed by inspecting *dictionaries* or *corpora*. Using *Wordnet* (Miller 1995), the ambiguity factor[15] would decrease down to about 1.44 (this factor seems to be fairly steady across languages such as English, Spanish and Italian), then we would obtain an order of complexity of O($1,44^n$). Slightly more optimistic results can be obtained with the *Brown corpus*: 40% of the words seem to be ambiguous and most of them are ambiguous between two *PoS*; then we could approximate the ambiguity factor up to 40%, obtaining an order of complexity of O($1,4^n$). This is clearly not enough yet for the problem to be tractable: analyzing a text would imply processing hundreds of words;[16] the exponential combinatory of the problem makes it impossible to find out a plausible solution in an acceptable time. Then, we should restrict the combinatorial domain across the input and/or use plausible clues to restrict the range of *ambiguity*.

One possible move is to consider the domain of *ambiguity resolution* to be restricted to a limited context: in fact, if the exponential *n* in the complexity function turns out to be a fixed number, the problem becomes tractable. But of course the context cannot be arbitrarily fixed (e.g. *n-grams* approach): the length of a grammatical phrase containing ambiguities can be arbitrarily long (theoretically infinite exploiting the complementation option), then fixing it once and for all would not be heuristic. It is also implausible to reduce the "structurally defined" context to the simple *local selection* as shown by the following contrasts:

(24) a. The [dogs] run ([D the] selects [N dogs])

b. Mary [dogs] me ([DP Mary] does not select any [V dogs])

The very same problem rises with *adverbials selection*: an adverb cannot *select* (in a technical sense) all the possible lower adverbials (that obviously can be present or not without affecting the grammaticality of the sentence). A more adequate solution could be to define the *phase* as the "largest context" within which an ambiguity can be solved: this would imply using the set of *dominance* relations projected according to *phase projection* to reduce the number of possible *PoS* associated to the set of *phonetic/semantic* features. Since a *phase* is dimensionally bounded in length (maximum number of *precedence* relations) and in depth (maximum number of *dominance*

[14] *Categories* (or *PoS*) have to be intended as "indices" (*Synsets* assuming the WordNet terminology) pointing to fully specified (in terms of features) items in the lexicon.

[15] ambiguity factor = $\frac{\text{synsets}}{\text{lexicalentries}}$

[16] With just 50 words to be disambiguated we could evaluate up to about 20 millions of possibilities.

relations)[17], the complexity order within each *phase* would be O($1, 4^{k+1}$) (where *1,4* is the most realistic ambiguity factor based on *dictionaries* and on *corpora*): the fundamental detail in this *complexity function* is that the exponent is, this time, a fixed number, virtually independent of the length *n* of the input. Note that opening more than one *phase* (from this perspective, any unselected argument, e.g. *preverbal subject*, represent a new phase opened within another *phase*) would produce an increase of the complexity order of O($1, 4^{p(k+1)}$) where *p*, the number of open *phases*, could grow boundlessly, in principle, leading quickly to intractability. This is however a welcome result, since a degradation of the human linguistic *performance* is reported in many processing data when the subjects have to parse/generate sentences with certain ambiguous structures that clearly show a difficulty in "keeping unsolved ambiguities" (e.g. "buffalo" sentences). Thus, the more *phases* we open (at the same time) the more difficult the problem will be as empirically expected.

2.2 The complexity of Long Distance Dependencies

Considering a *parsing* perspective[18] and a brute force algorithm, finding out which *dominance* relations have to be associated to a given set of *precedence* relations given in input has, at least, the complexity order of O(2^{n-1}), where *n* is the length of the input (namely the number of words in the sentence): this is because among *n* items we should define *n-1* relations at best (the minimum number of relations that would make a tree of *n* leafs, fully connected) and any of these relations can be ambiguous about the projecting head (A<B or B<A). Complexity rises, if we consider simple *movement* (let us put aside for the moment *cyclic movement*): at worse any item could have been potentially moved out from any lower (*C-commanded*, *selected*) position, the complexity order of the problem increases up to O($2^{(n^2-n)/2}$): this is because, potentially, any element could establish a dominance relation with any other element that follows it: e.g. with *4* elements {A, B, C, D} we could have *6* possible dominance relations {A-B, A-C, A-D, B-C, B-D, C-D}; with *5* elements {A, B, C, D, E} we could have *10* possible dominance relations {A-B, A-C, A-D, A-E, B-C, B-D, B-E, C-D, C-E, D-E}... then $\frac{((n-1)+1)\times(n-1)}{2}$. Complexity rises again (boundlessly this time), considering that empty heads[19] can enter *dominance* relations and there is no way to determine how many empty heads could be present, in principle, in a sentence of length *n*.

This is clearly not a satisfactory result, since the growing rate of any of these func-

[17] Given a fixed hierarchy of *licensors* features (Cinque 1999–2002), a *phase* contains at worst **k** *functional elements* (or *licensors*), exactly **1** projecting lexical element (by definition of *phase*). Remember then, that by assumption (Pesetsky 1982) each lexical head selects at most **3** *ph(r)ases* (subject, object and indirect object); this determines the maximum growth of the progression composed by the number of the (selected) phases which will be projected.

[18] I will not consider in these pages the *generation* problem, which is however much more "easy" than the *parsing* one: linearizing a set of dominance relations in a phase by phase procedure is straightforward once we have the *Linearization Principle* and a rigid order predicted among licensors features. A non trivial problem to be discussed is how to retrieve the exact set of dominance relations which belong to the phase, but this very same problem affects Chomsky's *numeration* idea too.

[19] At least in terms of π features.

tions would make the problem quickly intractable. Once again intractability in *parsing* can be tacked by adopting the idea of *phase* then working out the (22.c) factor: we assumed before (§1.4) that *movement* can be detected by the presence of an element that is not *selected* (that is, an element which is not selected by a previously processed lexical element, according to the *Linearization Principle*, (9)); in this case, this element would stand in "memory" as long as another element *selects* it.

Following Chomsky (1999), let us assume that if *movement* happens, it has to happen within the *phase*: then given a limited context corresponding to the *phase* boundaries, either we find the *selecting* element within it, so we can connect the moved object to its base position (projecting a set of *dominance* relations, according to *Phase Projection*, (17), that minimally satisfy this *selection* necessity), or we do not find any *selecting* element, then the expectations to find a *selecting* sister for the unselected element will be projected onto the *lower (selected) phase*[20] and so on. The properties of the intermediate traces[21] strengthen the idea that these traces on the *edge* (Chomsky 1999) of the *phase* serve as a sort of "memory refresh" (*phase balance*, Felser 2001), that is, the undischarged elements within the previous *memory buffer* are allowed to enter the "phase numeration" (then being part of the next selected *phase* processing) in order to make *Long Distance Dependencies* possible. This intuition produces a remarkable reduction of *complexity* (as shown in table 1): in fact, for any *moved* element, we would have only two possible landing site positions within a *phase* to be evaluated (one *left-edge* position, used as a memory refresh or one *selected* position). Then for any *phase* the number of *dominance* relations required would be, at worst, $2^{2k)}$ (in case anything would have been moved to the licensors field of the *phase*).[22] The complexity order of the problem, considering any *dominance* as ambiguous, would be O(2^{2k}). Opening a new *phase* before having closed the previous one, again, leads to a duplication of the number of possible *dominance* relations and so on as long as new *phases* are opened. The real order of the problem is then O(2^{p2k}) with *p* representing the number of open *phases* at the same time. The relation between the *length of the input* (*n*) and this function is expressed in terms of *phases*, since any *phase* represents a chunk of this input; in particular, the number of *lexical items* in *n* would determine the number of *phases* (which could be *n*, at worst). Note that *Discontinuous Constituency Dependencies* within a *phase* would not produce any remarkable effect on the *complexity* of the problem, while *discontinuous constituency relations* among *phases* would increase the *complexity* of the problem in a linear way if any *phase* is closed before the next one starts. On the other hand, there is an exponential increase of *complexity* any time we open a *phase* before having closed the previous one.

This expected increase of the *complexity* (parallel to the degradation in processing)

[20] Leaving an intermediate trace on the "left-periphery" of this lower *phase*, as discussed in Chesi (2004), Bianchi, Chesi (2005).

[21] They are not *selected* positions; they are available for *binding*, *reconstructions effects*; they are not triggered by any apparent satisfaction of semantic requirements, they are in fact accounted for, in a *bottom-to-top* perspective, by purely formal features.

[22] **k** is the number of functional features present in the current phase (which potentially can legitimate movement to their specific position), **1** lexical head, minus **1**, provide that for linking *n* elements we would need *n-1* relations.

functions	N° *of relations* *n*=6 (*p*=2, *k*=2)	*to be evaluated* *n*=9 (*p*=3, *k*=2)
Brute force $2^{(n^2-n)/2}$	$\simeq$ 32K	$\simeq$ 68.000M
Nested Phases 2^{p2k}	256	4096
Linear Phases $p \cdot 2^{2k}$	32	64

Table 1: Comparison among functions w.r.t. input length (assume each phase to be composed by 2 licensors features/positions and 1 lexical head).

related to the *movement* of elements across *open phases* exactly predicts *center embedding effects* (Chomsky 1965) and *strong island conditions* (Huang 1982, Bianchi, Chesi 2005).

3 Conclusion

In this paper I proposed a formalization of the notion of *phase* (Chomsky 1999) providing some arguments in favor of the necessity of including this component as part of the *Structure Building Operations* within a *Minimalist Grammar* formalism based on Stabler's 1997 proposal. Moreover, I assumed that the formalization of the other *Structure Building Operations merge* and *move*, also has to differ from Stabler 1997 in terms of *directionality*: the standard *Bottom-to-Top* derivation assumed in Chomsky's work has been show to be problematic from many perspectives (Chesi 2004) then it has be replaced by a *Top-to-Bottom*, *Left-to-Right* perspective, extending Phillips' 1996 original approach. Within this perspective, formalizing performance tasks such as (*parsing* and *generation*) has been shown to be a very useful tool in order to highlight the essential parameters to calculate the complexity function of a generalized algorithm that has to make an effective use of this *competence model*, pointing out that the formalized notion of *phase* and the directionality of the *move* operation produce a (psycholinguistically plausible) remarkable complexity reduction in both *ambiguities* and *long distance dependencies* resolution.

References

Bianchi, V. and Chesi, C.(2005), Phases, left branch islands and computational nesting, *Proceedings of the 29th PENN LINGUISTICS COLLOQUIUM*.

Boeckx, C. and Hornstein, N.(2004), Movement under control, *Linguistic Inquiry* **35**(3), 431–452.

Chesi, C.(2004), *Phases and Cartography in Linguistic Computation: toward a Cognitively Motivated Computational Model of Linguistic Competence*, PhD thesis, University of Siena.

Chomsky, N.(1995), *The Minimalist Program*, MIT Press, Cambridge (MA).

Chomsky, N.(1999), *Derivation by Phase*, MIT Occasional Papers in Linguistics, no. 18. Cambridge (MA).

Chomsky, N.(2001), *Beyond Explanatory Adequacy*, MIT Occasional Papers in Linguistics, no. 20. Cambridge (MA).

Cinque, G.(1999), *Adverbs and Functional Heads: A Cross-linguistic Perspective*, Oxford University Press.

Cinque, G.(2002), *The Structure of DP and IP. The Cartography of Syntactic Structures, Vol.1*, Oxford University Press.

Felser, C.(2001), Wh-copying, phases and successive cyclicity, *Essex Research Reports in Linguistics*.

Harkema, H.(2001), *Parsing Minimalist Languages*, PhD thesis, UCLA.

Huang, J.(1982), *Logical relations in Chinese and the theory of grammar*, PhD thesis, MIT.

Kayne, R.(1994), *The Antisymmetry of Syntax*, MIT Press, Cambridge (MA).

Kayne, R.(2002), *Pronouns and their antecedents*, Blackwell, Oxford.

Michaelis., J.(1998), *Derivational Minimalism is Mildly Context-Sensitive*, Springer Verlag, Berlin.

Miller, G. A.(1995), Wordnet: A lexical database for english, *Communications of the ACM*.

Nunes, J.(2004), *Linearization of Chains and Sideward Movement*, MIT Press, Cambridge (MA).

Papadimitriou, C.(1994), *Computational Complexity*, Addison-Wesley, Reading (MA).

Pesetsky, D.(1982), *Paths and Categories*, PhD thesis, MIT.

Phillips, C.(1996), *Order and Structure*, PhD thesis, MIT.

Shieber, S., Schabes, Y. and Pereira, F.(1995), Principles and implementation of deductive parsing, *Journal of Logic Programming*.

Stabler, E.(1997), Derivational minimalism, *Lecture Notes in Computer Science* **1328**, 68–98.

Starke, M.(2002), Against specifiers, *Abstract for TILT 2002*.

Between VP Adjuncts and Second Pole in Dutch

A corpus based survey regarding the complements between VP adjuncts and second pole in Dutch

Tim Van de Cruys

K.U.Leuven

Abstract

In Dutch, verbs are situated at fixed places in the sentence. Those places are called the first and second pole. VP adjuncts seem to function as some kind of pivot place in between these poles. This article investigates, by means of corpus research in the Spoken Dutch Corpus (CGN), which elements are intervening between these VP adjuncts and the second pole. Attention is particularly paid to the reasons and principles that make elements end up between VP adjuncts and second pole. First of all, these elements will often be syntactically and semantically linked to the main verb. Secondly, the functional sentence perspective will be important for the placement of elements before or behind the VP adjuncts. The results will show that the functional sentence perspective is one of the main information dividing principles in Dutch sentences. The functional sentence perspective is then implemented in Head-Driven Phrase Structure Grammar, extending Van Eynde's theory about Argument Realization in Dutch. Being able to handle focus information in an adequate way is important for contemporary issues such as coreference resolution. A better understanding of the principles that order the complements of the Dutch verb will also be helpful in correctly analyzing and parsing Dutch sentences.

1 Concepts

1.1 The Structure of Dutch Clauses

The Dutch grammar *ANS* (Haeseryn, Romijn et al. 1997) describes a Dutch main clause on the basis of a first and second pole, occupied by the verbs. The *Mittelfeld*, in between these two poles, contains three parts. The central part of the Mittelfeld is occupied by different kinds of VP adjuncts.

The structure of a Dutch subclause is quite different. The first pole is occupied by a conjunction, that connects the subclause to the main clause. The actual subclause starts with the Mittelfeld. All the verbs are put on the second pole.

	1st sentence position	**1st pole**	**MITTELFELD** *1*	*2 VP Adjuncts*	*3*	**2nd pole**	**last sentence position**
a	Ik	heb	Jan	gisteren	een boek	gegeven	
	I	have	Jan	yesterday	a book	given	
	I've given a book to Jan yesterday						
b	–	(...dat)	ik Jan	gisteren	een boek	gegeven heb	
	–	(...that)	I Jan	yesterday	a book	given have	
	...that I've given a book to Jan yesterday						

Table 1: The structure of a Dutch main clause (a) and subclause (b)

The actual position in which constituents end up, depends on all kinds of different ordering principles. We will only have a look at the ordering principles that are important for the position between VP adjuncts and second pole.

1.2 Between VP Adjuncts and Second Pole

There are two main reasons why elements are ending up between VP adjuncts and second pole. First of all, some constituents are **inherently connected to the main verb**. Sentence (1) gives an example. The predicative complement *groen* needs to be put between the VP adjuncts and the second pole. A sentence with the predicative complement between first pole and VP adjuncts, as in (2), is ill-formed.

(1) Ze hebben dat hekje gisteren groen geverfd.
they have that fence yesterday green painted
'They have painted that fence green yesterday.'

(2) * Ze hebben dat hekje groen gisteren geverfd.
they have the fence green yesterday painted

Secondly, the **functional sentence perspective** makes the most informative elements end up between VP Adjuncts and second pole. The ANS states that the VP adjuncts in the Mittelfeld function as some kind of pivot place: elements that are less informative appear before the VP adjuncts, more informative elements appear behind them. Sentence (3) gives an example. Sentence (4), where the new information is put between first pole and VP adjuncts, is highly questionable.

(3) Ik zal je morgen een boek geven.
I will you tomorrow a book give
'I will give you a book tomorrow.'

(4) ? Ik zal je een boek morgen geven.
I will you a book tomorrow give

It should be noted that the position between VP adjuncts and second pole is not the only position that is signaling focus information. This can also be the case with first sentence position. The first sentence position, which is the normal position for the subject in the main clause, is signaling focus information if it is taken by a constituent other than the subject, as in (5).

Also, the last sentence position often attracts extra attention, as in (6), although there are other reasons why a constituent might end up in this position.

(5) Een boek zal ik je morgen geven.
A book will I you tomorrow give
'I will give you a book tomorrow.'

(6) Ik heb gisteren een boek gegeven aan Jan.
I have yesterday a book given to Jan
'I've given a book to Jan yesterday.'

But in this paper, we will mainly focus on the position between VP adjuncts and second pole.

2 Methodology

The corpus research was carried out in the syntactically annotated part of CGN (Spoken Dutch Corpus). As has been indicated by van der Wouden et al. (2003), the CGN is a new resource for research into contemporary spoken Dutch that is well suited for carrying out statistical research in order to shed some light on certain linguistic issues. Only the Flemish Dutch part was used, as the Northern Dutch part was not available at the time of research. This corpus contained 42479 sentences, of which only the clauses with VP adjuncts and second pole were retained. The remainder consists of 3879 main clauses and 3309 subclauses. This corpus was searched with the syntactic search program TIGERSearch.

A statistical approach was taken in investigating the constituents between VP adjuncts and second pole. A number of queries were developed, that determined, for all complements of the verb (subject, direct object, indirect object, ...), in how many cases they end up between VP adjuncts and second pole. This number is then compared to the other possible places in which the complements can appear. This gives the following possibilities:

- Main clause:
    1. First sentence position
       Aan Jan *heb ik gisteren een boek gegeven*
    2. Between first pole and VP adjuncts
       Ik heb ***Jan*** *gisteren een boek gegeven*
    3. Between VP adjuncts and second pole
       Ik heb dat boek gisteren ***Aan Jan*** *gegeven*
    4. Final sentence position (extraposition)
       Ik heb dat boek gisteren gegeven ***aan degene die het graag wilde hebben***
- Subclause
    1. Before VP adjuncts
       ... dat ik ***Jan*** *gisteren een boek gegeven heb*
    2. Between VP adjuncts and second pole
       ... dat ik dat boek gisteren ***aan Jan*** *gegeven heb*
    3. Final sentence position (extraposition)
       ... dat ik dat boek gisteren gegeven heb ***aan degene die het graag wilde hebben***

3 Results

3.1 Subject

Table 2 gives the percentages of the different subject positions in a Dutch clause. In the main sentence, the subject can appear in three places: in first sentence position, between first pole and VP adjuncts, and between VP adjuncts and second pole. The results show that the subject appears only in few cases between VP adjuncts and second pole. In the majority of cases, the subject comes before the VP adjuncts (96.40%).

The results of the subclause are similar. The subclause lacks a first sentence position, but about 90% of the subjects appear before the VP adjuncts.

	main clause		**subclause**	
position	**n**	**%**	**n**	**%**
1st sentence position	2565	68.99%	–	–
1st pole - VP adjuncts	1019	27.41%	2486	89.88%
VP adjuncts - 2nd pole	120	3.23%	247	8.93%
extraposition	14	0.37%	33	1.19%
total	3718	100.00%	2766	100.00%

Table 2: The position of the subject

The interpretation of these results is quite straightforward: the functional sentence perspective is responsible for the distribution of the subject. The subject is usually a known entity, to which an unknown attribute is assigned (7). In passive sentences, however, there are some cases in which the subject can appear between VP adjuncts and second pole. This is the case if there is a pronoun which anticipates the subject (8), or if an adjunct acquires first sentence position (9). The subject then gets the focus of the sentence.

(7) Ik heb gisteren een koffie gedronken.
I have yesterday a coffee drunk
'I have drunk a coffee yesterday.'

(8) Er wordt ook wijn gedronken.
there is also wine drunk
'Wine is also drunk.'

(9) In de krant zijn toen veel spellingsbijlagen verschenen.
in the newspaper are then many spelling supplements published
'Many spelling supplements have been published in the newspaper at that moment.'

Note that, in order to put the subject in focus position, it needs to be placed between VP adjuncts and second pole. This is the only possibility to give focus to the subject,

because the first sentence position is the normal, unmarked position of the subject. For all the other complements, first sentence position (topicalisation) is a marked position, and hence attains focus.

3.2 Indirect Object

Table 3 gives the results of the indirect object. Clearly, the indirect object occurs mostly between first pole and VP adjuncts: in about 3 out of 4 cases in both main clause and subclause.

	main clause		**subclause**	
position	**n**	**%**	**n**	**%**
1st sentence position	7	5.22%	–	–
1st pole - VP adjuncts	100	74.63%	27	72.97%
VP adjuncts - 2nd pole	13	9.70%	7	18.92%
extraposition	14	10.45%	3	8.11%
total	134	100.00%	37	100.00%

Table 3: The position of the indirect object

The functional sentence perspective is again responsible for the distribution of the indirect object over the different positions in the clause. But this does not explain why there are more indirect objects that appear before the VP adjuncts. Upon examining the data a bit closer, an explanation comes up: the majority of the clauses with an indirect object is built according to the structure *first pole + personal pronoun + VP adjuncts (+ direct object) + second pole*, as in sentence (10). So in the majority of cases, the indirect object consists of a personal pronoun (a known entity) that does not bear the focus of the sentence. Hence, it is not put in focus position. If the indirect object is put into first sentence position or between VP adjuncts and second pole, it clearly bears the focus of the sentence, as in (11) and (12).

(10) Ik zal je meteen een voorbeeld geven.
I will you immediately an example give
'I will give you an example immediately.'

(11) Aan Jan zal ik dat boek morgen geven.
to Jan will I that book tomorrow give
'I will give that book to Jan tomorrow.'

(12) ... dat ik dat boek morgen aan Jan zal geven.
that I that book tomorrow to Jan will give
'... that I will give that book to Jan tomorrow.'

3.3 Direct Object

Table 4 gives the results of the direct object's position. The four positions are possible, but most of the direct objects end up in the Mittelfeld. The number of direct objects that is put before the VP adjuncts and behind the VP adjuncts is about the same.

	main clause		**subclause**	
position	**n**	**%**	**n**	**%**
1st sentence position	244	12.10%	–	–
1st pole - VP adjuncts	797	39.51%	492	40.39%
VP adjuncts - 2nd pole	737	36.54%	590	48.44%
extraposition	239	11.85%	136	11.17%
total	2017	100.00%	1218	100.00%

Table 4: The position of the direct object

Again, the functional sentence perspective is responsible for the position of the direct object. Compare sentences (13) and (14).

(13) Ik heb dat boek gisteren aan Jan gegeven.
I have that book yesterday to Jan given
'I've given that book to Jan yesterday.'

(14) Ik heb Jan gisteren een boek gegeven.
I have Jan yesterday a book given
'I've given that book to Jan yesterday.'

(15) * Ik heb een boek gisteren aan Jan gegeven.
I have a book yesterday to Jan given

In sentence (13), the direct object *dat boek* is presented as a known entity, while the indirect object *aan Jan* gets the focus. Sentence (14) gives the opposite situation: the indirect object *Jan* is known, but it is not known that *een boek* has been given to him.

The fact that the functional sentence perspective really does play an important role, is again proven by sentence (15): introducing an unknown object before the VP adjunct sounds awkward to the native speaker of Dutch.

3.4 Prepositional Complement

Sentence (16) is an example of a normal prepositional complement. But when discussing the prepositional complement (as well as the locative/directional complement in the next section), we need to take into account an extra particularity. In Dutch it is possible to split up a prepositional complement, if the head of the prepositional complement is a pronoun.[1] The pronoun is then put before the VP adjuncts, while

[1]This particular construction will be coined *discontinuous prepositional complement*, as opposed to *full prepositional complements*.

the preposition comes after the VP adjuncts (17). Moreover, it is not possible to put the pronoun between VP adjuncts and second pole when referring to inanimate objects (18). This is again a clear indication that the functional sentence perspective plays an important role.

(16) Ze hebben gisteren weer over voetbal gepraat.
they have yesterday again about soccer talked
'They have been talking about soccer again yesterday.'

(17) Ze hebben er gisteren weer over gepraat.
They have there yesterday again about talked
'They have been talking about it again yesterday.'

(18) * Ze hebben gisteren weer over het gepraat.
They have yesterday again about it talked

	main clause		**subclause**	
position	**n**	**%**	**n**	**%**
1st sentence position				
full	17	3.91%	–	–
discontinuous	35	8.05%	–	–
1st pole - VP adjuncts				
full	15	3.45%	12	3.93%
discontinuous	110	25.29%	44	14.43%
VP adjuncts - 2nd pole	126	28.97%	123	40.33%
extraposition	132	30.34%	126	41.31%
total	435	100.00%	305	100.00%

Table 5: The position of the prepositional complement

Table 5 gives the results of the prepositional complement. The results show that full prepositional complements mainly end up after the VP adjuncts. It seems that prepositional complements are either full and end up between VP adjuncts and second pole, or they are discontinuous, with the preposition between VP adjuncts and second pole, and the pronoun between first pole and VP adjuncts. There seems to be a strong link between the verb and its preposition, so that it needs to be realized near the verb. But when the prepositional contains known information, this conflicts with the functional sentence perspective. This is why the prepositional complement is split up, with the preposition realized near the verb (between VP adjuncts and second pole), and the pronoun moved between first pole and VP adjuncts.

With regard to these conflicting principles, sentence (19) is particularly interesting. It contains the referring pronoun *hem*, which is normally ending up between first pole and VP adjuncts due to the functional sentence perspective. This is shown by sentence (20), which expresses a similar meaning. Nevertheless, a sentence like (21) is questionable, because the preposition *over* is inherently connected to the main verb.

(19) Ze hebben gisteren weer over hem gepraat.
they have yesterday again about him talked
'They have talked about him again yesterday.'

(20) Ze hebben hem gisteren weer uitvoerig bediscussieerd.
they have him yesterday again ample discussed about
'They have talked a lot about him again yesterday.'

(21) ? Ze hebben over hem gisteren weer gepraat.
they have about him yesterday again talked

3.5 Locative/Directional Complement

The locative/directional complement subsumes all complements that are designating a place or a direction. They are either prepositional (22) or adverbial (23). Again, discontinuous complements are possible (24).

(22) Hij is dan naar dat eiland gezwommen.
he is then to that isle swum
'He has swum to that isle then.'

(23) We zijn toen huiswaarts gekeerd.
we are then towards home turned
'We went home then.'

(24) We gaan er bomen op planten.
we go there trees on plant
'We're going to plant trees on it.'

	main clause		**subclause**	
position	**n**	**%**	**n**	**%**
1st sentence position				
full	26	5.96%	–	–
discontinuous	7	1.61%	–	–
1st pole - VP adjuncts				
full	59	13.53%	71	16.95%
discontinuous	37	8.49%	33	7.88%
VP adjuncts - 2nd pole	261	59.86%	288	68.74%
extraposition	46	10.55%	27	6.44%
total	436	100.00%	419	100.00%

Table 6: The position of the locative/directional complement

Table 6 gives the results of the locative/directional complement. They are quite different to the results of the prepositional complement: full locative/directional complements appear more between first pole and VP adjuncts compared to the prepositional complement. Discontinuous complements are possible, but appear to a lesser extent compared to the prepositional complement. Most locative/directional complements end up between VP adjuncts and second pole.

The results might be explained as follows: the preposition is not inherently connected to the main verb, so that full complements can appear before the VP adjuncts. But the locative/directional complement most of the times contains focus information, so that it needs to be realized between VP adjuncts and second pole. Hence, there are also less discontinuous complements. So in this category, the functional sentence perspective plays an important role again.

3.6 Predicative Complement

The results of the predicative complement (table 7) are very straightforward. The majority of predicative complements ends up between VP adjuncts and second pole.

	main clause		**subclause**	
position	**n**	**%**	**n**	**%**
1st sentence position	13	4.13%	–	–
1st pole - VP adjuncts	9	2.86%	26	4.66%
VP adjuncts - 2nd pole	275	87.30%	510	91.40%
extraposition	18	5.71%	22	3.94%
total	315	100.00%	558	100.00%

Table 7: The position of the predicative complement

The explanation is simple: predicative complements are inherently connected to the main verb. They need to come obligatorily between VP adjuncts and second pole (except for some special cases like topicalisation). Compare sentence (25) and (26).

(25) Dat zal wel genoeg zijn.
that will well enough be
'That should be enough.'

(26) ? Dat zal genoeg wel zijn.
that will enough well be

The predicative complements occurring between first pole and VP adjuncts are due to spoken language characteristics, as in (27): normally, the modifier *uiteraard* comes before the predicative complement, but in spoken language one might modify the utterance 'on the fly', after having already uttered the predicative complement. Such a syntactic construction is not used in written language.

(27) Dat mag de Nederlandse tekst uiteraard zijn.
that may the Dutch text of course be
'That may of course be the Dutch text.'

For a more detailed discussion of the predicative complements, see Van Eynde (this volume).

4 Implementation in HPSG

Van Eynde (this volume) presents an HPSG theory to capture the different sentence positions in Dutch. Instead of using the popular classification of arguments (SUBJ, SPR and COMPS), a difference is made between arguments that need to be realized near the verb (COMPS) and arguments that can be separated from the verb by VP adjuncts (L-ARGS). This theory provides an adequate basis to capture the conclusions that have been deducted from the corpus research. Van Eynde offers a description of the inherently connected arguments, that are dependent on semantic and syntactic factors. I will focus on the description of pragmatics, i.e. the functional sentence perspective, in HPSG.

4.1 Capturing the Functional Sentence Perspective

The way of coding focus information into HPSG is based on the approach of Engdahl and Vallduví (1996). They provide a theory to capture the functional sentence perspective in English and Catalan.

$$\left[\text{SYNSEM|LOCAL|CONTEXT|INFO-STRUCT}\left[\begin{array}{ll}\text{FOCUS} & \langle[\ldots]\rangle\\ \text{LINK} & \langle[\ldots]\rangle\end{array}\right]\right]$$

Figure 1: Functional sentence perspective structure in HPSG

Figure 1 shows where focus information is coded in the HPSG structure. An extra attribute INFO-STRUCT, which contains the attributes LINK and FOCUS, is included in the CONTEXT attribute. The attribute LINK contains the complements that 'link' the sentence to former sentences or to known information. They are the known entities, about which something is said. The FOCUS attribute contains the new, informative information. Both FOCUS and LINK can take lists of values.

The corpus research has proven that the arguments between first pole and VP adjuncts contain the LINK information, while the arguments between VP adjuncts and second pole contain the FOCUS information[2]. Now that we have a way to code the different positions in HPSG (the division between COMPS arguments and L-ARGS arguments), the next step is to design a *Focus Realization Principle* which assigns the correct INFO-STRUCT values according to the position in the sentence. Figure 2 shows what this principle should look like.

[2]Next to first sentence position and last sentence position, as explained in 1.2.

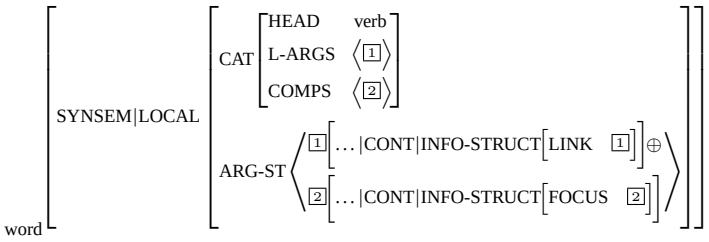


Figure 2: Focus Realization Principle

The information about word order (which determines the focus information) is available in the feature structure of the verb, in the L-ARGS and COMPS list. The Focus Realization Principle makes sure that the arguments which are link or focus, also get this characteristic coded into their feature structures. This is brought about by adding the information to the ARG-ST-list: arguments on the L-ARGS-list become link, arguments on the COMPS-list become focus. Note the fact that the value of LINK and FOCUS is equal to the sign itself.[3]

The Focus Realization Principle makes sure that the various arguments of the verb signal the right focus information. Now we only have to make sure that this information is passed on to the mother nodes, so that the final root node will also contain the correct focus information. This is done by the *Focus Inheritance Principle* in (28).

(28) **Focus Inheritance Principle**
The INFO-STRUCT value of the mother node is equal to the different INFO-STRUCT values of the child nodes.

This way, the focus information of the sentence is put together correctly.

4.2 An Example

Figure 3 shows the analysis of sentence (29).

(29) ... dat [ik hem]$_L$ gisteren [een boek]$_F$ gaf.
that I him yesterday a book gave
'... that I gave him a book yesterday'

The verb *gaf* has in this example two arguments *ik* and *hem* on the L-ARGS-list, and one argument *een boek* on the COMPS-list. When the VP is built up (according to the Argument Realization Principle), the Focus Realization Principle makes sure that the correct focus information is distributed over the various arguments. This way, *een*

[3]Engdahl and Vallduví note correctly that in this way, semantic information as well as phonological and syntactic information is marked as focus. It would be more correct to make the LINK and FOCUS values equal only to the semantic information of the sign. To keep a clear view, this approach is not elaborated, but we're assuming that it is done this way.

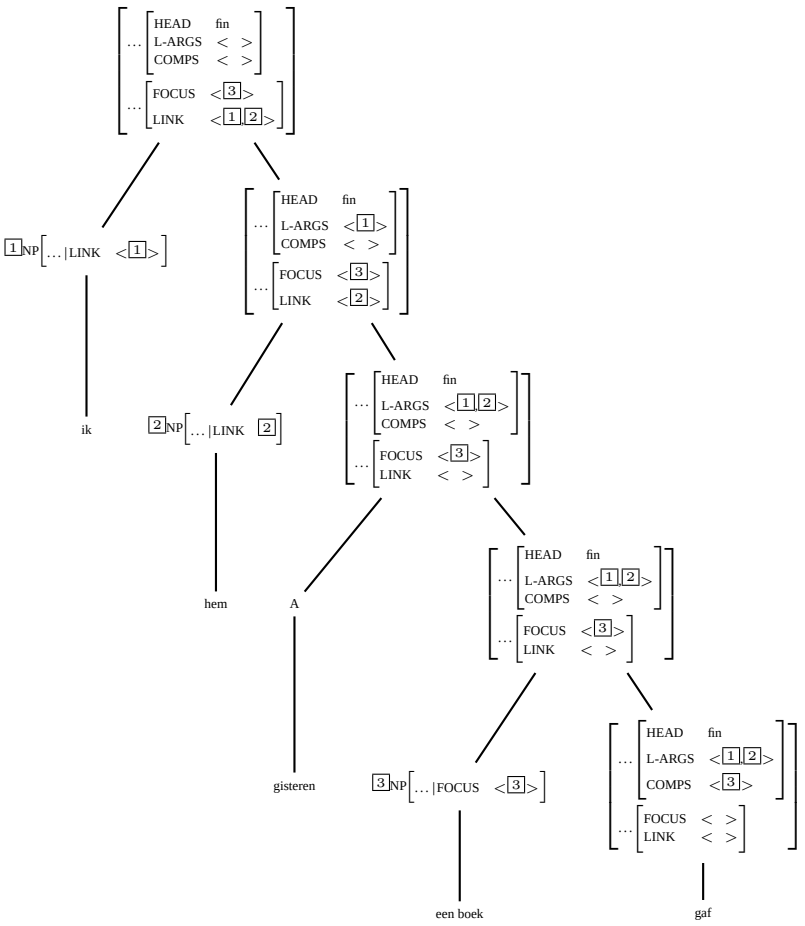


Figure 3: The focus structure of a subclause

boek signals that its meaning must be focus, and *ik* and *hem* signal that their semantic information must be linked to other known information. At the same time, the Focus Inheritance Principle makes sure that the focus information is passed on to the mother nodes, so that the root node of the clause contains the correct focus information.

5 Conclusion and Further Research

This paper has investigated the distribution of the various complements to the verb in a Dutch clause, and the reasons and principles that are responsible for the different distributions of these complements. It has become clear that, for certain complements, semantic (and syntactic) principles play an important role. Predicative complements need to appear close to the main verb because they are semantically linked to it. Also, the preposition of prepositional complements is closely linked to the verb. But these principles alone are not sufficient to explain the distribution of the various complements. The corpus research has clearly indicated that, for other complements, pragmatic principles play an equally important role. Complements that do not have a fixed position in the sentence are distributed according to the functional sentence perspective: unknown, informative information is put behind the VP adjuncts, while known information, that links the unknown information to the speaker's world, appears before the VP adjuncts.

Van Eynde (this volume) provides an HPSG implementation of the semantic and syntactic principles that are important in a Dutch clause. My paper has focused on an implementation of the pragmatic functional sentence perspective. It has been shown by the corpus research that this principle needs to implemented in the grammar, to be able to describe the formation of Dutch clauses in an adequate way.

At the same time, the description of the pragmatic principles in a Dutch clause is not yet complete. This paper has mainly investigated the position between VP adjuncts and second pole, leaving aside the other focus positions such as first and last sentence position. Also, it needs to be investigated in which way the focus information stemming from word order combines with the focus information that is conveyed by prosodic cues. These topics are to be investigated to get a complete view of how pragmatic principles influence the design of Dutch clauses.

It is only when these pragmatic principles have been investigated and added to the grammar, that a Dutch clause can be analyzed to its full extent.

References

Engdahl, E.(1999), Integrating pragmatics into the grammar, *in* L. Mereu (ed.), *Boundaries of Morphology and Syntax*, John Benjamins, Amsterdam.

Engdahl, E. and Vallduví, E.(1996), Information Packaging in HPSG, *Working Papers in Cognitive Science*.

Eynde, F. V.(this volume), Argument realization in an SOV language.

Haeseryn, W., Romijn, K. et al.(1997), *Algemene Nederlandse Spraakkunst*, Martinus Nijhoff uitgevers/Wolters Plantyn, Groningen/Deurne.

König, E., Lezius, W. and Voormann, H.(2003), *TIGERSearch 2.1 User's Manual*, University of Stuttgart, Stuttgart.

Sag, I. and Wasow, T.(1999), *Syntactic Theory. A Formal Introduction*, CSLI Publications, Stanford.

van der Wouden, T., Schuurman, I., Schouppe, M. and Hoekstra, H.(2003), Harvesting Dutch trees: Syntactic properties of spoken Dutch, *in* T. Gaustad (ed.), *Computational Linguistics in the Netherlands 2002. Selected Papers from the Thirteenth CLIN Meeting*, Rodopi, Amsterdam/New York, pp. 129–141.

Argument Realization in Dutch

Frank Van Eynde

Centrum voor Computerlinguistiek - Universiteit Leuven

Abstract

The treatment of argument realization is rather straightforward for a language like English, but for a language with relatively free word order, such as Dutch, it is a complex matter. It is not surprising then that the devices which are commonly used to deal with it show a high degree of computational complexity. They typically include movement, as in transformational grammar, or the dissociation of order-in-the-representation from the surface order, as in certain types of monostratal grammar. For the purpose of natural language *description* these devices are certainly convenient, but for the purpose of natural language *processing* they are less attractive. For this reason, I propose an alternative treatment of argument realization, which is consistently monostratal and surface-oriented. Its cornerstone is the GENERALIZED ARGUMENT REALIZATION PRINCIPLE. It is a generalization of the Argument Realization Principle which is proposed in (Ginzburg and Sag 2000) to deal with English.

1 The argument realization principle

The lexical entries of argument taking words commonly include information about the number and the kinds of arguments which they select. Intransitive verbs, for instance, select one NP and transitive verbs two. This information is useful for syntactic and semantic processing, on condition that it is complemented with information on how the arguments are realized in sentences. For a language with a relatively rigid word order, such as English, the constraints on argument realization are relatively straightforward. In an active English clause, for instance, the first NP argument is realized as the subject and precedes the verb, whereas the other arguments follow the verb in a fixed order.

(1) a. They gave her a bike.
b. * Gave they her a bike.
c. * They her a bike gave.
d. * They gave a bike her.

To spell this out in formal terms HEAD-DRIVEN PHRASE STRUCTURE GRAMMAR employs two kinds of features. There is the ARG(UMENT)-ST(RUCTURE) feature, which specifies the syntactic and semantic properties of the arguments which a word selects, and there are the valence features SUBJ(ECT), SP(ECIFIE)R and COMP(LEMENT)S, which spell out how these arguments are realized. The link between them is defined by the ARGUMENT REALIZATION PRINCIPLE (Ginzburg and Sag 2000, 23).[1]

[1]Throughout the text, the boxed alphabetic characters stand for lists of objects, whereas the boxed integers stand for individual objects, usually of type *synsem*. $\oplus$ is a concatenation operation on lists.

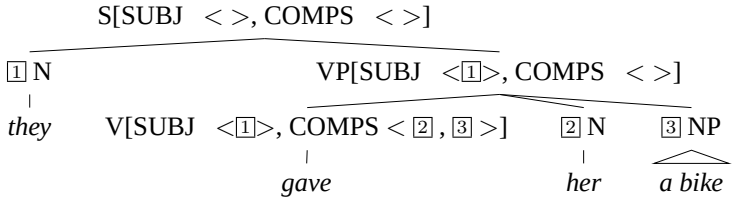
(2) *word* ⇒ $\begin{bmatrix} \text{ARG-ST} & \boxed{A} \oplus \boxed{B} \oplus \boxed{C} \\ \text{SYNSEM}\,|\,\text{LOC}\,|\,\text{CAT} & \begin{bmatrix} \text{SUBJ} & \boxed{A} \\ \text{SPR} & \boxed{B} \\ \text{COMPS} & \boxed{C} \end{bmatrix} \end{bmatrix}$

In words, the list of arguments is divided in three sublists; the members of the first sublist are realized as subjects (□*A*), those of the second one as specifiers (□*B*) and those of the third one as complements (□*C*). Subsets of words are associated with more specific constraints. Verbs, for instance, have a singleton SUBJ list and an empty SPR list, which means that they take one subject and no specifier (o.c., 22). When applied to the ditransitive *give*, this yields the following result.

(3) $\begin{bmatrix} \text{ARG-ST} & \langle \boxed{1}\,\text{NP}, \boxed{2}\,\text{NP}, \boxed{3}\,\text{NP} \rangle \\ \text{SYNSEM}\,|\,\text{LOC}\,|\,\text{CAT} & \begin{bmatrix} \text{SUBJ} & \langle \boxed{1} \rangle \\ \text{SPR} & \langle\,\rangle \\ \text{COMPS} & \langle \boxed{2}, \boxed{3} \rangle \end{bmatrix} \end{bmatrix}$

The first argument is realized as a subject and the remaining arguments as complements. These requirements interact with the constraints on headed phrases: the combination of the verb with its complements is subsumed by the type *head-comps-phrase*, yielding a VP, and the combination of the VP with the subject is subsumed by the type *head-subj-phrase*, yielding a fully saturated clause, as in (4).

(4)



The distinction between the subject and the complements is not only motivated by their position (preverbal vs. postverbal), but also by other factors. The subject, for instance, can be separated from the verb by a modifier, but the complements cannot.

(5) a. They never gave her a bike.
b. * They gave never her a bike.
c. * They gave her never a bike.

Similarly, in the case of raising, it is only the subject of the embedded verb which is affected, not any of its other arguments.

(6) a. She seems to like her bike.

b. * She seems her bike to like.

In sum, the distinction between subject and complements is easy to draw in a language like English. Dutch, by contrast, is a language with relatively free word order, and this seriously complicates the treatment of argument realization. First, there is the variation between SVO and SOV order, which implies that the complements can either follow or precede the verb.

(7) a. Ze gaven haar een fiets.
they gave her a bike

b. ... dat ze haar een fiets gaven.
... that they her a bike gave

Second, the mutual order of the arguments is not fixed. In English the indirect object NP invariably precedes the direct object NP, and this is also a possibility in Dutch, as illustrated by (7), but the alternative order is also possible and sometimes obligatory, for instance, when the direct object is a weak pronoun, such as *het* 'it'.

(8) We geven het haar morgen.
we give it her tomorrow

Third, VP-adjuncts are routinely interleaved with the complements and often separate them from the verb, both in SVO and SOV clauses.

(9) a. Ze gaven haar elk jaar een nieuwe fiets.
they gave her every year a new bike

b. ... dat ze haar elk jaar een nieuwe fiets gaven.
... that they her every year a new bike gave

Fourth, raising is not restricted to subjects, as will be illustrated in section 5. The modeling of argument realization is, hence, considerably more complex for Dutch than it is for English. To deal with this complexity there are basically two approaches.

One approach is to separate the issues of argument realization and linear order. More specifically, one can use the same general constraints on argument realization as for English, and add a proviso that they only hold for a level of syntactic structure in which the constituents do not necessarily occur in the surface order. This approach is typical of multistratal models, such as those of transformational grammar, which postulate a level of structure, at which the relation between the selected arguments and their realization is straightforward (deep structure or d-structure), but at which the order of the constituents does not correspond to their linear order in the clause. The resulting gap is bridged by movement operations, such as raising and scrambling. A similar approach is taken in certain variants of monostratal grammar. Dowty's distinction between tectogrammatical and phenogrammatical structure, for instance, has—in this respect—roughly the same function as the distinction between deep and surface structure, and has inspired a strand of HPSG in which the order of the words in the syntactic representation does not necessarily correspond to their order in the PHONOLOGY value. This approach was pioneered in (Reape 1994) and further developed in (Kathol 2000).

The alternative approach is to modify the constraints on argument realization. For instance, in order to account for the interleaving of arguments and adjuncts in Dutch, (Van Noord and Bouma 1994) adds the adjuncts to the COMPS lists of the verbs, so that there is no longer any need for scrambling operations. The proposal to be presented in this paper is another instance of this approach, but develops it in another direction. Instead of obliterating the distinction between arguments and adjuncts, I will preserve it and argue in favor of making some finer-grained distinction between different kinds of arguments, depending on how they are positioned with respect to the adjuncts. The approach is similar in spirit to the one proposed for Norwegian in (Hellan and Haugereid 2004).

2 The generalized argument realization principle

The cornerstone of the proposal is the GENERALIZED ARGUMENT REALIZATION PRINCIPLE (GARP).[2]

(10) *word* $\Rightarrow$ $\begin{bmatrix} \text{ARG-ST} & \boxed{A} \oplus \boxed{B} \\ \text{SYNSEM}\,|\,\text{LOC}\,|\,\text{CAT} & \begin{bmatrix} \text{L-ARGS} & \boxed{A} \\ \text{COMPS} & \boxed{B} \end{bmatrix} \end{bmatrix}$

In words, the arguments which some given word selects are realized as either complements or l(eft)-arguments.[3] The complements are those arguments which are adjacent to their head in SOV clauses and which cannot be separated from their head by means of adjuncts. The l(eft)-arguments, by contrast, are the arguments which can be separated from their head by adjuncts. This can be captured in the formula [L-Arg* - Adj* - Comp* - Head], in which * is the Kleene star. When applied to (11), it turns out that the subject and the indirect object are l-arguments, since they both precede the adjunct *elk jaar*.

(11) … dat ze haar elk jaar een nieuwe fiets gaven.
… that they her every year a new bike gave

If the head takes the leftmost position, as in SVO clauses, the formula reads as follows: [Head - L-Arg* - Adj* - Comp*]. Notice that it is only the position of the head which changes; the mutual order of the arguments and the adjuncts is the same as in the head-final sequence. In (12), for instance, the pronoun *haar* 'her' is an l-argument, just as in (11).

(12) Ze gaven haar elk jaar een nieuwe fiets.
they gave her every year a new bike

The term *left-argument* is not only chosen because it applies to the arguments which remain after the complements have been subtracted but also because the l-arguments

[2] This is a preliminary version. The final version will be given in section 5.

[3] For the reasons given in (Van Eynde 2003) I do not employ a separate valence feature for the selection of specifiers. I, hence, drop SPR from the inventory of valence features.

invariably occur to the left of the complements, both in head final and head first constructions.

As compared to the familiar distinction between complements and subjects the one between complements and l-arguments is more general. More specifically, the former can be derived from the later if one adds the constraint that the L-ARGS list contain at most one member. This extra constraint may well be appropriate for some languages, such as English, but for a language like Dutch it is too strict. At the same time, the differentiation between complements and l-arguments in Dutch is not entirely without constraints either. In fact, there is a number of factors which steer the partitioning. One such factor concerns the properties of the selecting words; another one concerns the properties of the arguments themselves, and a third factor concerns the constraints on information packaging. The first two factors are discussed and illustrated in sections 3 and 4 of this paper; the third one is explored in (Van de Cruys this volume).

3 Predicate selectors

To illustrate the role of the selecting words in the partitioning of the arguments I take the verbs which select a predicate. Such verbs take at least two arguments. One is the predicate, which denotes a property, and the other one is an NP which denotes the entities to which the property is attributed. The adjectival predicate of the copula in (13), for instance, denotes a property which is attributed to the individual denoted by *Patrick*.

(13) ... dat Patrick elke morgen voor zes uur wakker is.
* ... dat Patrick wakker elke morgen voor zes uur is
'... that Patrick is awake before six every mornning.'

As illustrated by the starred clause, the predicate cannot be separated from the verb by the temporal adjuncts. This implies that it must be realized as a complement. To express this I introduce a specific lexical type for the words which select a predicate, to be called *pred(icate)-sel(ector)*, and assign it the following properties:

(14)
$$\left[\begin{array}{l} \textit{pred-sel} \\ \text{ARG-ST} \quad \boxed{X} \oplus \left\langle \boxed{1} \right\rangle \oplus \left\langle \boxed{2} \right\rangle \oplus \boxed{Y} \\ \text{SS} \mid \text{LOC} \mid \text{CAT} \left[\begin{array}{l} \text{L-ARGS} \quad \boxed{X} \oplus \left\langle \boxed{1}\ \text{NP}_i \right\rangle \\ \text{COMPS} \quad \left\langle \boxed{2} \left[\text{LOC} \mid \text{CONT} \mid \text{NUCL} \mid \text{THEME} \quad i \right] \right\rangle \oplus \boxed{Y} \end{array}\right] \end{array}\right]$$

In words, the property denoting argument of a predicate selector must be realized as a complement (2) and the argument to which the property is attributed must be realized as a left argument (1).

Besides the fact that it must be realized as a complement, there are no syntactic constraints on the predicate. Its part of speech, for instance, need not be adjectival, but can also be nominal, prepositional or adverbial, as in (15).

(15) a. ... dat hij later dokter wordt.
* ... dat hij dokter later wordt
'... that he'll become a doctor later.'

b. ... dat zijn geduld nu wellicht op geraakt.
* ... dat zijn geduld op nu wellicht geraakt
'... that his patience is probably running out by now.'

c. ... dat hij nog niet weg is.
* ... dat hij weg nog niet is
'... that he is not away yet.'

The predicate may also be phrasal, as illustrated in (16).

(16) a. ... dat zijn rapport blijkbaar [vol fouten] staat.
* ... dat zijn rapport [vol fouten] blijkbaar staat
'... that his report is apparently full of mistakes.'

b. ... dat die boeken volgens hem niet [voor kinderen] zijn.
* ... dat die boeken [voor kinderen] volgens hem niet zijn
'... that those books are not for children according to him.'

In these clauses the predicate denotes a relation between the referent of its external argument and the referent of its own complement. The adjective *vol* in (16a), for instance, denotes a relation between the referents of *zijn rapport* and *fouten*, and the preposition *voor* in (16b) denotes a relation between the referents of *die boeken* and *ons*.

The only restriction on the predicate is a semantic one: it must denote a relation (with arity equal or greater than one) in which the role of theme is realized by the external argument. The relevance of this constraint is illustrated by (17).

(17) a. ... dat ze zichzelf niet meer zijn.
... that they themselves not more are
'... that they are not themselves anymore.'

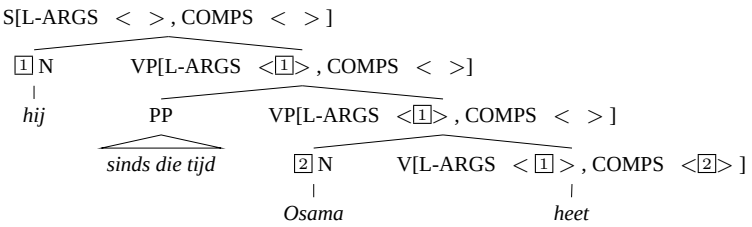
b. ... dat dat Osama niet is.
... that that Osama not is
'... that that is not Osama.'

In contrast to the predicate nominal in (15a), the reflexive pronoun in (17a) and the proper noun in (17b) precede the negation marker, which implies that they are l-arguments, rather than complements. This, however, is not a violation of the *pred-sel* constraint, for since the NPs in (17) denote individuals, rather than properties or relations, they are not subsumed by the constraint. Further evidence for the semantic nature of the constraint is provided by (18).

(18) ... dat hij sinds die tijd Osama heet.
* ... dat hij Osama sinds die tijd heet.
'... that he is called Osama since then.'

In this clause, *Osama* does not denote an individual, but a name, and since names can be attributed to individuals, they are subsumed by the *pred-sel* constraint. This accounts for the fact that *Osama* must precede the temporal adjunct. Employing the distinction between COMPS and L-ARGS, (18) is analyzed as follows:

(19)



The verb takes the name as its complement (2) and the subject as its left argument (1). The lower VP has an empty COMPS list and combines with the PP adjunct yielding the higher VP which then combines with the l-argument *hij* 'he'. In this example, the distinction between complement and left argument coincides with the distinction between complement and subject. This, however, is not always the case. In (11), for instance, the indirect object sides with the l-arguments, and the same holds for the reflexive pronoun and the proper noun in (17).

As for the external argument of the predicate, the only restriction is that it be an NP with a referential index. This NP can be the subject, as in the examples above, but it can also be the direct object, as in (20).

(20) ... dat ze hem elke morgen voor zes uur wakker maakt.
* ... dat ze hem wakker elke morgen voor zes uur maakt.
'... that she wakes him before six every mornning.'

In this clause, the adjectival complement of the causative verb *maakt* 'makes' denotes a property which is attributed to the individual denoted by *hem* 'him'. To model this instance of object oriented predication we do not need to make any changes to the constraint on predicate selectors, since (14) foresees that the external argument of the predicate may be preceded by other arguments (X). In the case of *maken*, X is a list which contains one NP (the subject); in the case of the copula, it is the empty list.

As the reader can verify, the constraint also foresees the possibility that the predicate complement is followed by other arguments (Y). The relevance of this addition will become clear in the next section.

4 Minor arguments

It is not only the properties of the argument selecting words which influence the partitioning between complements and l-arguments. Equally important are the properties of the selected arguments themselves. To demonstrate this I will now discuss two classes of arguments which share the property that they invariably consist of a single word, i.e. the particles and the weak pronouns.

4.1 Particles

Particles are closely tied to the verb, both semantically and syntactically. In the orthography this is sanctioned by the convention to treat them as verbal prefixes, as in (21a).

(21) a. ... dat ik ze onmiddellijk opbel.
... that I her immediately upcall
'... that I call her immediately.'

b. Ik bel ze onmiddellijk op.
I call her immediately up
'I call her immediately.'

c. ... dat ik ze onmiddellijk op zal bellen.
... that I her immediately up will call
'... that I will call her immediately.'

In spite of its intuitive appeal, though, the affixal treatment of the particles is not satisfactory, since it is only applicable in those cases in which the particle happens to immediately precede the verb. If this condition is not fulfilled, as in (21b) and (21c), the particle must be treated as a separate word.[4] As a consequence, since the contribution of the particle is the same in (21a) as it is in (21b) and (21c), it makes more sense to treat it as a separate syntactic unit, also when it is adjacent to the verb.

To model the distribution of the particles, I start from the assumption that they are on the ARG-ST list of the verbs to which they are related. This is motivated a.o. by the fact that their presence may influence the rest of the verb's argument structure. *Lachen* 'laugh', for instance, is intransitive when it is not accompanied by a particle, but when it is accompanied by the particle *uit*, it is transitive. Conversely, *zien* 'see' is transitive when used without a particle, but when it is accompanied by the particle *af*, it has the meaning of suffering and in that sense it is intransitive.

As for the realization of the particles, they invariably occur after the adjuncts, both in SOV and SVO clauses.

(22) a. ... dat ik ze onmiddellijk op zal bellen.
* ... dat ik ze op onmiddellijk zal bellen

b. Ik bel ze onmiddellijk op.
* Ik bel ze op onmiddellijk

This demonstrates that they are complements. To express this in formal terms I employ the following constraint:

[4]Some of the particles are homophonous to prefixes, but it is easy to distinguish them, since the prefixes cannot be separated from the verb, whereas the particles can. Compare, for instance, the prefix in *Hij overdrijft altijd* 'he always exaggerates' with the homophonous particle in *Hij steekt de straat over* 'he crosses the street'.

(23)
$$\left[\begin{array}{l} \textsc{arg-st} \quad \textit{nelist} \oplus \left\langle \boxed{1} \left[\textsc{loc}\,|\,\textsc{cat} \left[\begin{array}{ll} \textit{minor} & \\ \textsc{head} & \neg\ \textit{p-noun} \end{array}\right]\right] \right\rangle \\ \\ \textsc{synsem}\,|\,\textsc{loc}\,|\,\textsc{cat}\,|\,\textsc{comps} \quad \boxed{X} \oplus \left\langle \boxed{1} \right\rangle \end{array}\right]$$

In words, the particles are the most oblique arguments. They are preceded by at least one other argument and must be realized as complements. Besides their rightmost position on the ARG-ST list, there is one other characteristic which distinguishes the particles from the other arguments, i.e. the fact that they are minor. The distinction between major and minor words, introduced in (Van Eynde 1999), differentiates the words which can take local dependents from those which cannot. The former are major and can head a branching XP; the latter are minor and lack this possibility. This differentiates a.o. the prepositions which are used as predicate complements from the homophonous prepositional particles.

(24) a. ... dat de voorraad nu [helemaal op] is.
... that the stock now [entirely up] is
'... that the stock is entirely finished now.'

b. * ... dat hij je morgen [helemaal op] zal bellen.
* ... that he you tomorrow [entirely up] will call

While the prepositional predicate in (24a) can take a modifying adverb and project a PP, the particle in (24b) cannot. Another difference between major and minor words is that the former can be stressed and topicalized, whereas the latter cannot. Compare, for instance, the predicate complement in (25a) with the particle in (25b).

(25) a. Op is het nog niet, maar we moeten nu wel zuinig zijn.
up is it not yet, but we must now economical be
'It is not finished yet, but we have to be economical now.'

b. * Op moet je hem niet bellen.
* up must you him not call

Since the major/minor distinction is applied to the values of CAT(EGORY), it is orthogonal to the part of speech distinction, and, hence, applicable to all parts of speech. This is the way it should be, since the particles can also take the form of adjectives, common nouns and adverbs.

(26) a. ... dat ze dit niet (*bijzonder) goed zal keuren.
... that she this not (*particularly) good will deem

b. ... dat het concert vorige week (*een) plaats had moeten vinden.
... that the concert last week (*a) place had must find
'... that the concert should have taken place last week.'

c. ... dat hij die bloemkool niet (*ver) weg had moeten geven.
... that he that cauliflower not (*far) away had must give
'... that he should not have given away that cauliflower.'

The impossibilty of adding a dependent to *goed*, *plaats* and *weg* demonstrates that they are minor, and the fact that they cannot be topicalized confirms this.

(27) a. * Goed zal hij het niet keuren.
* good will he it not deem

b. * Plaats zal het morgen vinden.
* place will it tomorrow find

c. * Weg zal hij het niet geven.
* away will he it not give

The only part of speech to which the particles cannot belong is the one of the pronouns.[5] A justification for this exclusion will be given in the next section.

If a verb takes both a predicate complement and a particle, the constraint requires that the former precede the latter. This follows from the fact that the particle is the rightmost member of COMPS in (23) and from the fact that the predicate complement can be followed by another complement in the *pred-sel* constraint, see $\boxed{Y}$ in (14). The correctness of this prediction is illustrated by (28).

(28) ... dat we hem vanmorgen dood aan hebben getroffen.
* ... dat we hem vanmorgen aan dood hebben getroffen
'... that we found him dead this morning.'

The adjectival predicate (*dood*) and the particle (*aan*) are both complements of *getroffen* and must occur in that order.[6]

4.2 Weak pronouns

Most of the Dutch pronouns come in two kinds. Besides the strong forms, such as *wij* 'we' and *dat* 'that', which have a clear vowel, there are the weak forms, such as *we* 'we' and *het* 'it', which have a mute vowel. Syntactically, the strong forms behave like major words, whereas the weak forms behave like minor words. The strong forms, for instance, can take local dependents, whereas the weak forms cannot, as illustrated by the following examples, quoted from (Model 1991, 287).

(29) Wij/*we allen hebben daar aan meegewerkt.
we all have that on collaborated
'We all have contributed to that.'

(30) Wij/we hebben allen daar aan meegewerkt.
we have all that on collaborated
'We have all contributed to that.'

[5] The part of speech *p-noun* contrasts with *c-noun*. The former subsumes all of the pronouns and most of the proper nouns; the latter subsumes the common nouns and a subset of the proper nouns. For a motivation of this distinction, see (Van Eynde 2003).

[6] The particles and the predicates are not the only arguments which must be realized as complements. Other such arguments are the NP objects of measure verbs, as in *vijf euro kosten* 'to cost five euro' and *drie kilo wegen* 'to weigh three kilos', and the arguments which are part of idiomatic expressions, as in *de benen nemen* 'to take the legs', i.e. 'to escape'.

Since verb-second clauses can only have one constituent before the verb, the NP in (29) must be branching and its head must, hence, be a nominal which can take local dependents. This accounts for the fact that the strong form *wij* can be used in this position, whereas the weak form *we* cannot. In (30), however, both forms can be used, for since the quantifier is in postverbal position, the subject NP is nonbranching and may, hence, also consist of a pronoun which cannot take any local dependents. The minor status of the weak pronouns is confirmed by the fact that they cannot be stressed or topicalized.

(31) Dat/*het zou ik niet doen.
that/*it would I not do
'That I wouldn't do.'

Returning now to the issue of argument realization, there is a conspicuous difference between the major and the minor pronouns. Both tend to precede the VP adjuncts, but while the major ones can also be realized after the adjuncts if they receive stress, the minor ones cannot.

(32) Ze willen nu eerst en vooral JOU/*je ondervragen.
they want now first and foremost YOU interrogate
'They first want to interrogate YOU now.'

Arguments which take the form of a minor pronoun must, hence, be realized as left arguments. In other words, they must be assigned to the L-ARGS list of the selecting word.

(33)
$$\left[\begin{array}{l} \text{ARG-ST} \quad \boxed{X} \oplus \left\langle \boxed{1} \left[\text{LOC}\,|\,\text{CAT} \left[\begin{array}{l} \textit{minor} \\ \text{HEAD} \quad \textit{p-noun} \end{array}\right]\right] \right\rangle \oplus \boxed{Y} \\ \\ \text{SYNSEM}\,|\,\text{LOC}\,|\,\text{CAT}\,|\,\text{L-ARGS} \quad \boxed{X} \oplus \left\langle \boxed{1} \right\rangle \end{array}\right]$$

There are no constraints on the relative position of the pronoun in the ARG-ST list: it may but need not be preceded by other arguments ($\boxed{X}$) and it may but need not be followed by other arguments ($\boxed{Y}$). There are also no constraints on the part of speech of the selector.

4.3 Summing up

Arguments which take the form of minor words are subject to some specific constraints: when they are pronouns, they must be realized as l-arguments, and when they belong to another part of speech they must be realized as complements.

5 Argument raising

In all of the examples discussed thus far the argument selecting word was a verb. Verbs, however, are not the only words which select syntactic arguments. Adjectives

and prepositions are argument selectors as well, and are, hence, also subsumed by the GENERALIZED ARGUMENT REALIZATION PRINCIPLE. This implies that their arguments can also be differentiated in complements, on the one hand, and left arguments, on the other hand. The PP argument of the adjective *bewust* 'aware' in (34), for instance, is separated from its head by the adverb *zeer* 'very', and the pronominal argument of the preposition *onder* 'under' in (35) is separated from its head by *vlak* 'right'.

(34) ... dat we ons van dat probleem [zeer bewust] waren.
... that we us of that problem [very aware] were
'... that we were very aware of that problem.'

(35) ... dat hij daar [vlak onder] stond.
... that he that [right under] stood
'... that he stood right under that.'

This demonstrates that they are both left-arguments. Moreover, the left arguments of the adjective and the preposition can also be separated from their head by adjuncts which belong to the verb, such as the negation *nog niet* 'not yet' in (36) and the temporal *toen* 'then' in (37).

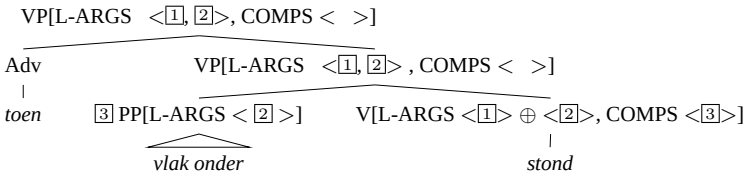
(36) ... dat we ons van dat probleem nog niet bewust waren.
... that we us of that problem still not aware were
'... that we were not yet aware of that problem then.'

(37) ... dat hij daar toen vlak onder stond.
... that he that then right under stood
'... that he then stood right under that.'

To model this, I assume that the L-ARGS requirements of the adjective, c.q. preposition, are inherited by the verb which selects the adjective, c.q. preposition. More specifically, I introduce a lexical type *l-arg-raiser* with the following properties:

(38)
$$\left[\begin{array}{l} \textit{l-arg-raiser} \\ \text{ARG-ST} \quad \boxed{X} \oplus \left\langle \left[\text{LOC} \mid \text{CAT} \mid \text{L-ARGS} \quad \boxed{A} \right] \right\rangle \oplus \boxed{Y} \\ \text{SYNSEM} \mid \text{LOC} \mid \text{CAT} \mid \text{L-ARGS} \quad \boxed{Z} \oplus \boxed{A} \end{array}\right]$$

The words which belong to this type add the L-ARGS requirements of their arguments to their own L-ARGS list. When applied to the VP in (37), this yields the following structure:

(39)


The verb takes the PP as its complement (3) and inherits its L-ARGS requirement (2), which is appended to its own L-ARGS list (1).[7] This analysis of l-arg raising also accounts for the contrasts in (40).

(40) a. ... dat ze het hier nog steeds niet gewoon is.
* ... dat ze hier nog steeds niet [het gewoon] is.
'... that she is still not used to it here.'

b. ... dat ze er toen nog niet tegen was.
* ... dat ze toen nog niet [er tegen] was.
'... that she was not yet against it.'

The pronominal arguments of the adjective *gewoon* 'used to' and the preposition *tegen* 'against' are realized by minor pronouns (*het* and *er*), and must, hence, be realized as l-arguments. Moreover, since the copula is an l-arg-raiser, the L-ARGS list of the predicate is appended to the L-ARGS list of the copula, which implies that the minor pronouns must precede the VP adjuncts.

A consequence of this treatment of argument raising is that a selector may have elements on its L-ARGS list which do not correspond to any of the elements on its ARG-ST list. The PP in (34) and the accusative pronouns in (35) and (40), for instance, are l-arguments of the verb, but do not figure on its ARG-ST list. This implies that we have to revise the formulation of the GENERALIZED ARGUMENT REALIZATION PRINCIPLE as follows:

(41) *word* $\Rightarrow$ $\left[\begin{array}{l} \text{ARG-ST} \quad \boxed{A} \oplus \boxed{B} \\ \text{SYNSEM}\,|\,\text{LOC}\,|\,\text{CAT}\left[\begin{array}{l}\text{L-ARGS} \quad \boxed{A} \oplus \boxed{X} \\ \text{COMPS} \quad \boxed{B}\end{array}\right] \end{array}\right]$

This version allows the L-ARGS list of a selector to include arguments which do not figure on its own ARG-ST list (X). This is not only useful to capture the intuition that the raised arguments have another status than the verb's own arguments, it also provides an account of the contrast between (42) and (43).

(42) ... dat hij elke morgen (door haar) wakker wordt geschud.
... that he each morning (by her) awake is shaken
'... that he is shaken awake (by her) every morning.'

(43) *... dat deze omgeving niet (door hem) gewoon geraakt wordt.
*... that this environment not (by him) used-to gotten is

As illustrated by (42), the external argument of an object-oriented predicate can become the subject of a passive clause. This can be made more specific in terms of the lexical rule which standard HPSG employs to model passivization. This rule maps the

[7] It may be worth stressing that this analysis does not involve any movement; the pronominal argument of *onder* 'under' is not moved out of the PP. Instead, the link between its surface position and its canonical (or semantically 'natural') position is defined in terms of the sharing of selection requirements. Constraints on movement, hence, become constraints on the sharing of selection requirements.

stem of a transitive verb onto its passive participle form and reorders its ARG-ST list in such a way that the NP in the first position (<NP_i, NP_j, ...>) is demoted to a more oblique position and realized as an optional PP complement (<NP_j, ..., (PP[P + NP_i]), ...>). An automatic consequence of this reshuffling is that the originally second argument becomes the first and gets realized as the subject. Turning now to (43), we can account for its ungrammaticality, if we assume that the NP *deze omgeving* is not on the ARG-ST list of the verb, for in that case it cannot be promoted to its first position. Interestingly, this assumption need not be stipulated anywhere, since it automatically follows from the fact that it is a raised argument, so that it only figures on the L-ARGS list of the verb and not on its ARG-ST list.

Summing up, the distinction between complements and left arguments is not only useful to model linear order in the VP, it also provides a way to identify those arguments which can undergo raising. For a language like English, these arguments only include the subject, since the other arguments are complements, but for a language like Dutch in which most of the arguments are left arguments, raising can be applied to a much wider range of arguments.

6 Conclusion

For a language with relatively free word order, such as Dutch, the relation between argument selection and argument realization is considerably more complex than for a language like English (section 1). The devices which are commonly employed to deal with this problem involve movement, as in transformational grammar, or the dissociation of order-in-the-representation from order-in-the-clause, as in certain types of monostratal grammar. While these devices are convenient for the purpose of language description, they are less appealing for the purpose of language processing. From a computational point of view it is more attractive to stick to the monostratal surface-oriented approach of early HPSG. This, I claim, is possible, if one draws the distinction between the different kinds of arguments in another way than is commonly done in current HPSG. Taking the surface order as a criterion, rather than as a distorting nuisance, I replace the distinction between complements and subjects with a more general distinction between complements and left arguments. The resulting treatment of argument realization is modeled in terms of the GENERALIZED ARGUMENT REALIZATION PRINCIPLE (section 2), supplemented with a number of constraints which apply to specific types of argument selecting words (section 3) and specific types of arguments (section 4). Further evidence for the generalized treatment of argument realization is provided by the phenomenon of raising (section 5).

References

Ginzburg, J. and Sag, I.(2000), *Interrogative Investigations*, CSLI, Stanford.

Hellan, L. and Haugereid, P.(2004), Norsource - an exercise in the matrix grammar-building design, NTU Trondheim.

Kathol, A.(2000), *Linear Syntax*, Oxford University Press.

Model, J.(1991), *Grammatische analyse. Syntactische verschijnselen van het Nederlands en Engels*, ICG Publications, Dordrecht.

Reape, M.(1994), Domain union and word order variation in German, *in* J. Nerbonne, K. Netter and C. Pollard (eds), *German in HPSG*, CSLI Publications, Stanford, pp. 151–197.

Van de Cruys, T.(this volume), Between VP adjuncts and second pole in Dutch, University of Leuven.

Van Eynde, F.(1999), Major and minor pronouns in Dutch, *in* G. Bouma, E. Hinrichs, G.-J. Kruijff and R. Oehrle (eds), *Constraints and Resources in Natural Language Syntax and Semantics*, CSLI Publications, Stanford, pp. 137–151.

Van Eynde, F.(2003), Morpho-syntactic agreement and index agreement in Dutch NPs, *in* T. Gaustad (ed.), *Computational Linguistics in the Netherlands 2002*, Rodopi, Amsterdam - New York, pp. 111–127.

Van Noord, G. and Bouma, G.(1994), Adjuncts and the processing of lexical rules, *Proceedings of the 15th International Conference on Computational Linguistics (Coling)*, University of Kyoto, Kyoto, pp. 250–256.

Accentuation of Adpositions and Particles in a Text-to-Speech System for Dutch

Nicole Grégoire

University of Utrecht

Abstract

In this paper I propose an accent placement algorithm that locates accents on adpositions and particles for the use in a Dutch text-to-speech (TTS) system. The algorithm is intended to be a refinement of the rule that accents only content words, which is used in most TTS systems. Before the algorithm is set up, I discuss when adpositions and particles are accented in Dutch. For this empirical research, I made use of the Spoken Dutch Corpus (CGN) as empirical material. The combination of part-of-speech, syntactic as well as prosodic information for approximately 125,000 words in the CGN made it possible to determine whether the accentuation of adpositions and particles depends on their syntactic use within a sentence, or on the syntactic use of other constituents in the same sentence. The proposed accentuation algorithm takes a dependency tree with part-of-speech information as input.

1 Introduction[1]

It is important that the speech generated by computers sounds as natural as possible. This leads to a more intelligible content and as a result it takes less effort for listeners to understand the meaning of an utterance and it will be easier for them to listen to synthetic speech. An important feature that contributes to the naturalness of the quality of speech is prosody. The word prosody refers to certain properties of speech, such as the location and duration of breaks between two parts of utterances, the duration of a syllable and the absence or presence of accents.

In this paper I will concentrate on one aspect of prosody, i.e. which words in an utterance are spoken with accent and which ones are spoken without accent. To correctly predict the location of accents in a sentence one needs a fully specified syntactic analysis and an interpretation of the utterance. Other helpful information is the context in which the sentence is uttered and the intention of the speaker. In general we cannot expect all this information to be available in a text-to-speech (TTS) system. Therefore, a very simple solution (1) has been proposed to approximate the correct generation of accentuation in a TTS system:

(1) Put accent only on content words but not on function words.

Content words are nouns, adjectives, adverbs and lexical verbs. Function words are articles, prepositions, pronouns, auxiliaries and conjunctions. The only information needed when a system uses this rule (1) is a distinction between content words and function words. For Dutch, this rule correctly predicts accent for 79% of the words in a sentence (Marsi et al. 2002). Though this is an impressive result with such a simple rule, it still means that on average three words in a sentence have an incorrect accent

[1] This introduction is based on Jan Odijk's inaugural lecture (Odijk 2003).

(assuming an average sentence length of 15 words), which shows that improvement is desirable.

The goal of this paper is to change rule (1) in such a way that accentuation in TTS systems improves, and that the speech output becomes more natural and intelligible. This study will focus on the accentuation of Dutch function words such as *in* ('in'), *op* ('on') and *naar* ('to'). These words can be used as adpositions[2] and particles. Examples of prepositions, which are adpositions that precede their complement, are given in (2) (the preposition is in italics). Particles are the separable part of separable compound verbs, such as *opbellen* ('to call (up)'), *aanstaren* ('to stare at'), *uitleggen* ('to explain'). These particles can occur separated from the verb, as in (3), where both the verb and the particle are in italics.

(2) a. Hij zet de bloemen *op* de tafel.
He puts the flowers on the table

b. Ik vlieg morgen *naar* Schotland.
I fly tomorrow to Scotland
'I will fly to Scotland tomorrow.'

(3) a. Ik *belde* hem *op*
I called him up
'I called him (up).'

b. Hij heeft altijd al met haar *uit* willen *gaan*.
He has always already with her out want go
'He had always wanted to go out with her.'

Odijk (2003) notices that words used as a particle are accented, whereas the same words used as an adposition are often not accented. Accentuation rule (1) does not distinguish this difference in accentuation, as both particles and adpositions belong to the group of function words.

The main question to be answered in this paper is given in (4).

(4) How can the accentuation of adpositions and particles be regulated automatically in a Dutch text-to-speech system?

I stated above that "the same words used as adpositions are often not accented", which implies that there are cases where adpositions are accented. In order to answer the question given in (4), a subquestion (5) will be answered.

(5) When are adpositions and particles accented in Dutch human speech?

In the next section I discuss when adpositions and particles are accented. In section 3 I present my accentuation algorithm. Section 4 concludes this paper.

[2] In most literature, the notion *preposition* is used instead of *adposition*. But in strict terms, a preposition is a word that precedes its complement. In this paper I also distinguish circumpositions, intransitive adpositions, stranded prepositions, and adpositions that are part of a pronominal PP (in Dutch a pronominal PP is called *voornaamwoordelijk bijwoord* and an example is *erop* in the sentence *ik plak het erop* ('I stick it on it')) and therefore I use the notion *adpositions* as a general term.

2 Accent placement on adpositions and particles

In my MA thesis (Grégoire 2004) a special chapter is devoted to the different uses of adpositions and adpositional phrases in Dutch. For each P[3] it was empirically tested whether it is accented in Dutch. To this end, the Corpus Gesproken Nederlands (CGN, Spoken Dutch Corpus)[4] was used as empirical material. The CGN contains approximately 125,000 prosodically annotated words. Besides prosodic information, part-of-speech (PoS) as well as syntactic information of each word is accessible. Within the syntactic information not only category labels are available, but also dependency labels. These dependency labels play an important role in the determination of accents on adpositions and particles.

As a starting point for this empirical investigation I used, amongst others, hypotheses formulated by Jan Odijk (2003 and personal communication) and adapted in such a way that they could be tested empirically using the data in the CGN. I formulated my findings by means of a set of claims that state that a P is accented under certain circumstances. The proposed accentuation algorithm is based on these claims. Because the main topic of this paper is to present the algorithm, I will not go into detail on how the empirical research was carried out, but just state the formulated claims in (6).

(6) a. The second part of a circumposition is accented if and only if its complement is not accented.

b. An intransitive adposition is always accented.

c. A particle is accented if and only if it is not directly adjacent to a focused argument.

d. A stranded preposition in an LD PP is accented if and only if it is not adjacent to a focused constituent.

e. A preposition in a pronominal LD PP is accented if and only if it is not adjacent to a focused constituent.

With LD PP is meant that the PP is a locative or directional complement, which can be recognized by the dependency label LD in the CGN. In the next section I will elaborate on these claims.

3 Accentuation of Ps in a TTS system for Dutch

3.1 Introduction

A TTS system generally deals with input text in two stages. In the first stage the input is linguistically analysed. The result is a phonetic representation of the utterance, which serves as input of the second stage, the speech synthesis, which involves

[3] I use the notion *Ps* to refer to all adpostions and particles.

[4] The *Spoken Dutch Corpus* is a database of contemporary Dutch as spoken by adults in the Netherlands and Flanders. The project is funded by the Flemish and Dutch governments and the Netherlands Organization for Scientific Research NWO. Its homepage is http://lands.let.kun.nl/cgn/ehome.htm.

converting this representation into a synthetic speech signal. In this paper I focus on converting an input text consisting of words into the same text in which the accents are marked, ignoring processes such as grapheme-to-morpheme conversion and speech synthesis.

The process I propose that generates an accented text from an unaccented text includes three tasks, viz.

1. part of speech tagging
2. syntactic parsing
3. accent placement

Merely PoS and syntactic information is not sufficient as input for accent placement, as predicting accent locations requires semantic and discourse information as well. Syntactic parsing in the CGN project involves — besides the assignment of category labels to each mother node — the assignment of dependency labels that denote the relation of a certain constituent with respect to another constituent dominated by the same mother node. This means that in the output of the syntactic annotation semantic information is visible to a certain extend. Because of this and because I assume that PoS tagging and syntactic annotation can be done automatically, I suggest that the applications used by the CGN project for PoS tagging and syntactic annotation should be used to perform task 1 and 2.

It must be taken into account that since both the PoS tags and the syntactic annotations were manually checked, we cannot just rely on the automatic output and that improvement of these applications is needed. I will discuss this point in the next section. In this section I assume that the output of automatic PoS tagging and syntactic annotation is 100% consistent and reliable.

The output of the CGN PoS tagger and syntactic annotator, which serves as the input for the accent placement task is a dependency tree such as Figure 1.
As was stated above, each mother node is assigned a category label (*c*-label: in the cylinder-boxes). Dependency labels (*d*-labels, in the square boxes) denote the relation of a certain constituent with respect to another constituent dominated by the same mother node. Each mother node contains at least the *d*-label HD (head).

In the remainder of this section I discuss the assignment of accents to a dependency tree such as Figure 1. I propose an algorithm for accent placement on adpositions and particles for the use in Dutch TTS systems. This algorithm is intended to be a refinement of the basic rule that simply assigns an accent to every content word.

The approach adopted to attain this goal starts from the focus and accentuation proposals in Marsi (2001). Although Marsi's rules are implemented in a concept-to-speech system,[5] whereas my proposals are for a text-to-speech system, I will show that using a dependency tree such as Figure 1 as input for the accentuation rules, Marsi's proposals can be adapted in such a way that they can predict the accent locations of adpositions and particles in a TTS system.

[5] In concept-to-speech systems, spoken output is generated on the basis of a text that has been produced by the system itself.

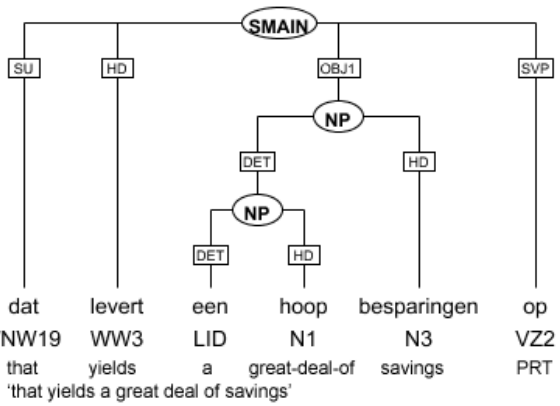


Figure 1: A dependency tree.

3.2 Accentuation algorithm

In this section I adapt Marsi's proposals so that it can take a dependency tree with part-of-speech information, such as Figure 1, as its input. In addition I propose a set of rules that locate accents on those Ps that are to be accented according to the claims formulated in the previous section. The accentuation algorithm I propose is presented in (7).

(7) **Accentuation algorithm**

1. Assign focus
2. Apply the Focus Projection Rules
3. Apply the Rules for Focusing Adpositions and Particles
4. Apply the Sentence Accent Assignment Rule

The input of the accentuation algorithm is restricted to a dependency tree that contains at least one verbal domain. A verbal domain is a constituent the head (HD) of which is a (finite or infinite) verb. Six verbal domains are distinguished in the CGN. The category label and a description of each verbal domain are given in Table 1. A verbal domain may be embedded in another verbal domain. The accentuation rules I propose are to be applied to each verbal domain.

3.2.1 Focus assignment

The first task of the accentuation algorithm is to assign focus (the feature FOC) to every content word. Since it is not so clear whether adverbs are content words or function words (Marsi 2001, 232), I follow Marsi in his decision not to focus adverbs. The part-of-speech tags — used in the CGN — in Table 2 are content words:

Table 1: Category labels of verbal domains in the CGN.

c-label	description
SMAIN	declarative sentence (verb second)
SSUB	subordinate clause (verb final)
SV1	verb first sentence
INF	infinitive clause
PPART	past participle clause
PPRESS	present participle clause

Table 2: Parts-of-speech distinguished in the CGN that are content words.

part-of-speech	abbreviation for	translation
ADJ	*adjectief*	adjective
N	*nomen*	noun
TW	*telwoord*	numeral
WW	*werkwoord*	verb

The focus assignment rule has no access to the given-new information of the sentence constituents. This means that every content word is focused, even if it conveys 'old' information. In Figure 1 the words *levert*, *hoop* and *besparingen* are content words according to their part-of-speech labels. In the dependency tree shown in Figure 2 the feature FOC is assigned to each content word.

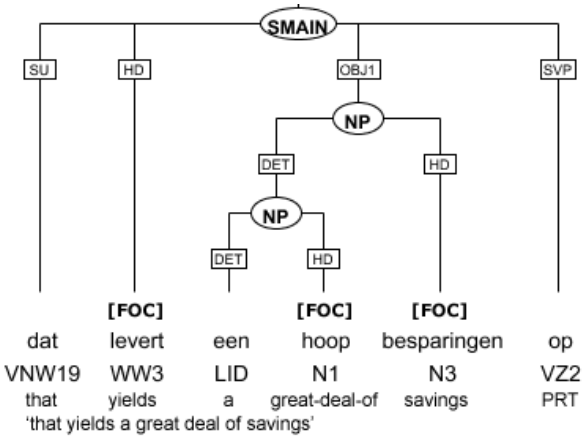


Figure 2: Assigning the [FOC]-feature to each content word.

3.2.2 Focus Projection Rules

The accentuation algorithm employs a set of Focus Projection Rules proposed by Marsi (2001, 217), based on Selkirk (1984, 1995):

(8) **Focus Projection Rules**

a. If the head of an NP is focused, then the NP is focused as well.

b. If the head of an AP is focused, then the AP is focused as well.

c. If the NP argument of a P is focused, then the PP is focused as well.

d. If the main verb in an S is focused, then the S is focused as well.

e. If all the conjuncts of a coordination are focused, then the coordination is focused as well.

According to these rules it is not sufficient if only the modifier of an NP, AP or S is focused. If, for example, only the N *London* but not the N *train* in the NP *the train to London* is focused, rule (8c) will project focus on the PP *to London*, but there is no rule in (8) that licences any further projection. This means that the NP is not focused, according to FPR. However, if the N *train* is focused, rule (8a) will project focus onto the whole NP.

Applying the Focus Projection Rules to the output of the focus assignment task results in Figure 3.

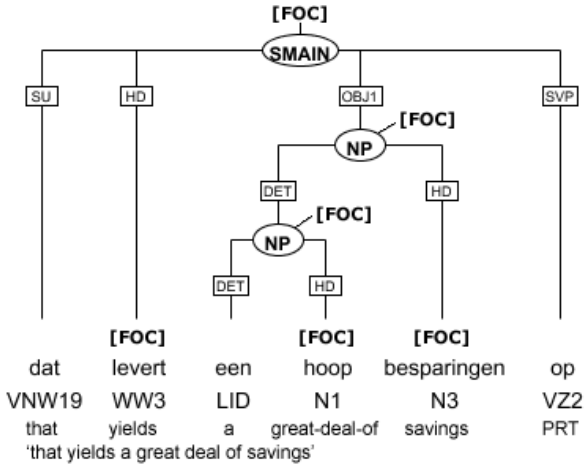


Figure 3: Applying the Focus Projection Rules.

3.2.3 Rules for Focusing Adpositions and Particles

In this section rules for focusing adpositions and particles are proposed. Not all Ps are to be focused. The claims that state that a P is accented under certain circumstances

are repeated in (9).

(9) a. The second part of a circumposition is accented if and only if its complement is not accented.

b. An intransitive adposition is always accented.

c. A particle is accented if and only if it is not directly adjacent to a focused argument.

d. A stranded preposition in an LD PP is accented if and only if it is not adjacent to a focused constituent.

e. A preposition in a pronominal LD PP is accented if and only if it is not adjacent to a focused constituent.

On the basis of these claims I propose the following sets of focus rules:

(10) **Focus Rules for Adpositions** (FRA)
Assign the feature [FOC] to each 'VZ2'[6] word if,

(i) it carries the *d*-label HDF and if the OBJ1 within the PP does not have the FOC-feature, or

(ii) 1. it is directly attached to the verbal domain without an intervening PP *c*-label, and
2. does not have the *d*-label SVP, or

(iii) it carries the *d*-label OBJ1 within a PP, or

(iv) 1. it carries the *d*-label HD within a PP, and
2. the OBJ1 within the PP has the PoS label 'VNW20' [7] or 'VNW15'[8].
3. this PP has the *d*-label LD, and
4. there is no constituent that is marked with the FOC-feature adjacent.

(11) **Focus Rule for Particles** (FRP)
Assign the feature [FOC] to a 'VZ2' word that has the *d*-label SVP, if there is no argument that is marked with the FOC-feature adjacent.

In addition I propose a rule for defocusing the verbal part of a particle verb.[9]

(12) **Defocusing Rule for Verbal parts of particle verbs** (DRV)
Delete the FOC-feature from the head of the verbal domain in which a word that has the PoS label 'VZ2' and the *d*-label SVP occurs.

[6]The PoS tag 'VZ2' is used in the CGN for final Ps.
[7]The PoS tag 'VNW20' is used for the R-pronouns *er*, *d'r*, *daar* and *hier*.
[8]The PoS tag 'VNW15' refers to the R-pronoun *waar* ('where'), which is attached to the PP by a secondary edge.
[9]In Grégoire (2004), no conclusions were drawn regarding the accentuation of the verbal parts of particle verbs. Therefore, I follow Marsi (2001) in his assumption that if a particle verb is accented, the accent goes to the particle.

As claimed, a preposition in a pronominal LD PP is accented if and only if it is not adjacent to a focused constituent. Pronominal PPs in the CGN do not belong to the adposition set, but are PoS tagged as 'BW' (abbreviation of *bijwoord* ('adverb')). Since there are many types of adverbs, which are not further subdivided in the CGN, a more refined classification is needed in order to distinguish the pronominal PPs from other adverbs. This problem can be solved by assuming that pronominal PPs can be recognised in the CGN by the lexicon and that they are split into two separate words that are PoS tagged and syntactically annotated as stranded prepositions. An example is given in Figure 4: the left tree shows how pronominal PPs are dealt with in the CGN and the right tree shows my suggestion on how the CGN should deal with pronominal PPs. Since the preposition in a pronominal PP is to be focused under the same circumstances a stranded preposition is focused under, no extra focus rule for pronominal PPs is needed.

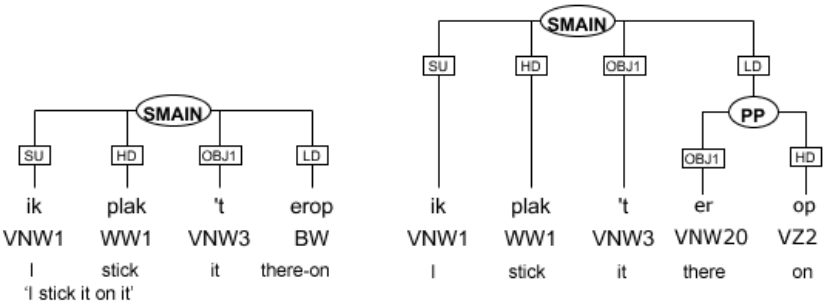


Figure 4: Pronominal PPs.

The output of FRA (i) is given in Figure 5 and 6. In Figure 5 the *heen* is assigned the

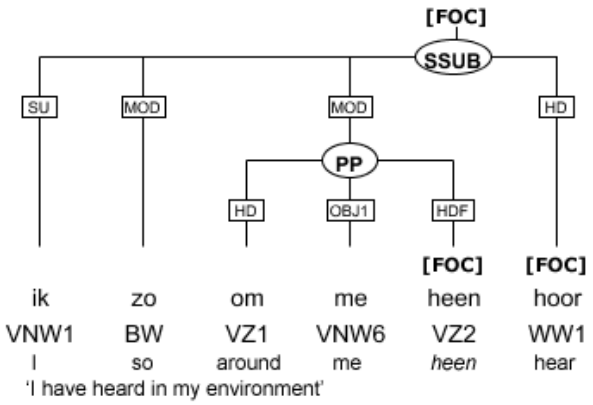


Figure 5: Output of FRA (i): *heen* is focused.

feature [FOC], because it has the PoS tag 'VZ2' and the *d*-label HDF, and the OBJ1 within the PP does not have the FOC-feature. Although *toe* in Figure 6 has the PoS tag 'VZ2' and the *d*-label HDF, FRA does not assign the FOC-feature to it, because the OBJ1 within the PP is focused.

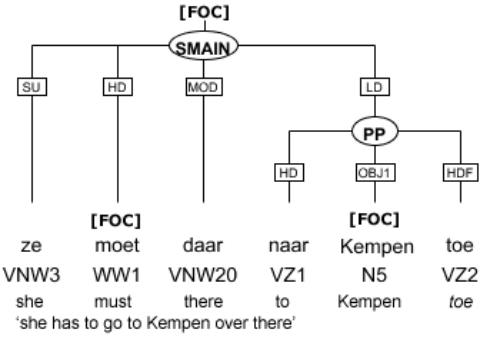


Figure 6: Output of FRA (i): *toe* remains unfocused.

An illustration of the output of FRA (ii) is shown in Figure 7.

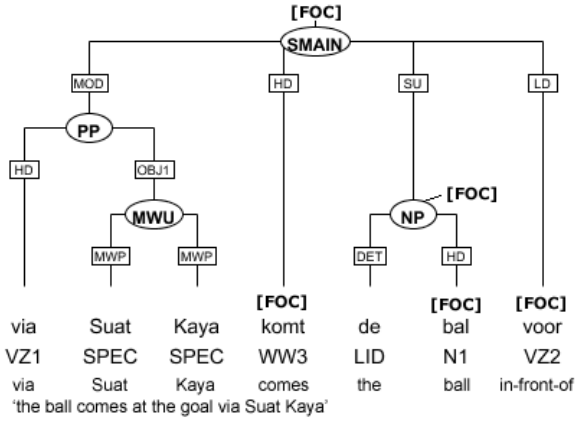


Figure 7: Output of FRA (ii): *voor* is focused.

An example of the output of FRA (iii) is given in Figure 8.

FRA (iv) is illustrated in Figure 9–11. In Figure 9 *bij* is PoS tagged as 'VZ2', it is the head of the PP, the OBJ1 within that PP is a 'VNW20', and this PP had the *d*-label LD. However, *bij* is not focused, because there is an adjacent focused constituent, in this case the subject NP *zo'n vragenlijst*. In Figure 10 the stranded preposition *in* is focused, because no focused constituent is adjacent. In Figure 11 the pronominal PP is annotated as proposed and since no focused constituent is adjacent to the LD PP, the

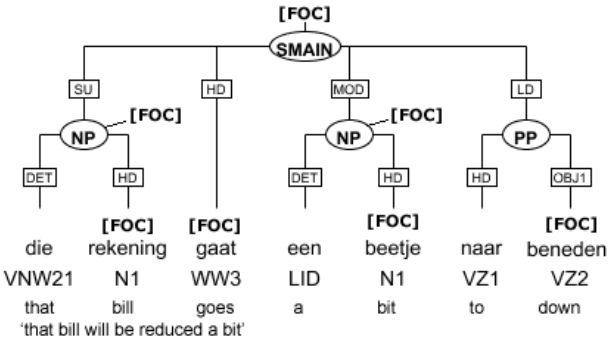


Figure 8: Output of FRA (iii): *beneden* is focused.

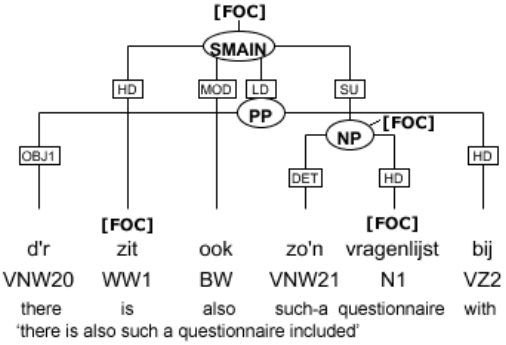


Figure 9: Output of FRA (iv): *bij* remains unfocused.

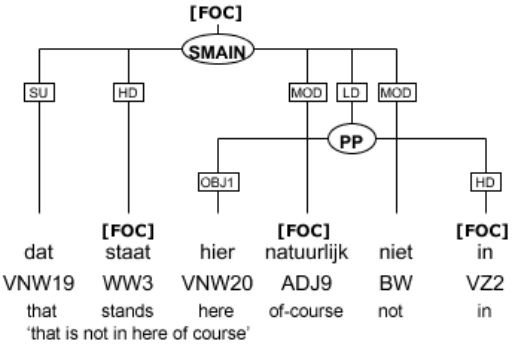


Figure 10: Output of FRA (iv): *in* is focused.

preposition *op* in this PP is assigned the FOC-feature.

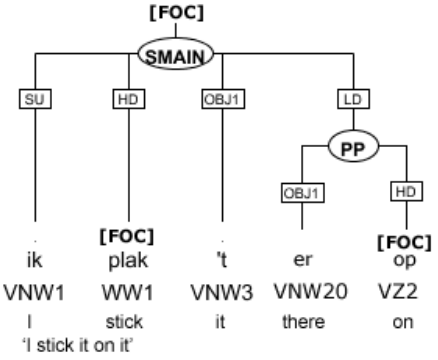


Figure 11: Output of FRA (iv): *op* is focused.

The output of FRP and DRV is illustrated in Figure 12 and 13. In both Figure 12 and 13 DRV deletes the FOC-feature on the head of the verbal domain, because a word — in Figure 12 *op* and in Figure 13 *uit* — that has the PoS label 'VZ2' and the *d*-label SVP occurs in the same domain. In Figure 12 the particle *op* remains unfocused, because a focused argument — the OBJ1 NP *een hoop besparingen* — is adjacent, whereas in Figure 13 the FOC-feature is assigned to the particle *uit*, since no focused argument is adjacent.

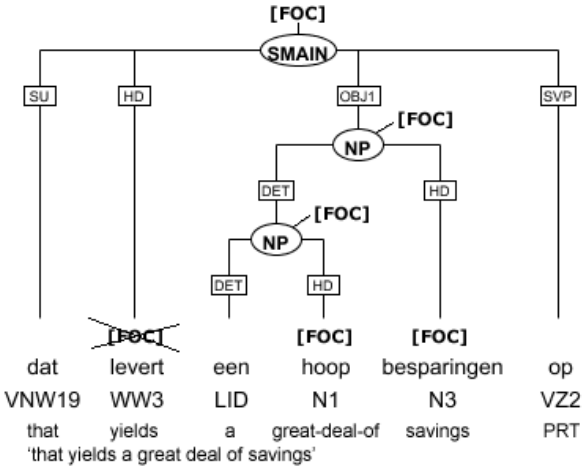


Figure 12: Output of FRP and DRV: The [FOC]-feature on *levert* is deleted and *op* remains unfocused.

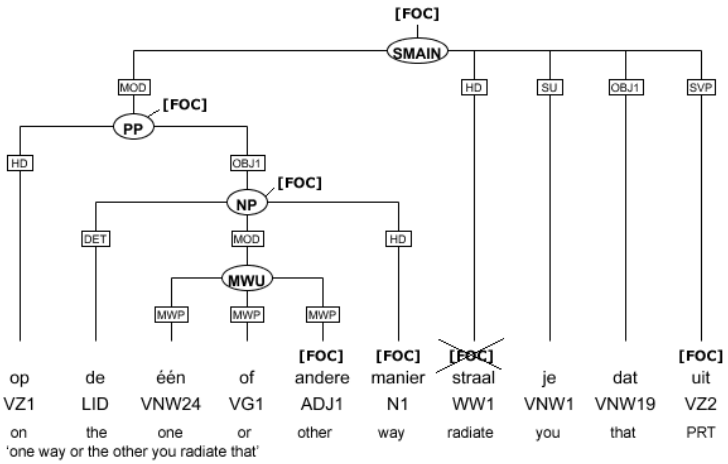


Figure 13: Output of FRP and DRV: The [FOC]-feature on *straal* is deleted and *uit* is focused.

3.2.4 Sentence Accent Assignment Rule

The outputs of the preceding section constitute the input of the Sentence Accent Assignment Rule (Gussenhoven 1992):

(13) **Sentence Accent Assignment Rule** (SAAR)
If focused, every predicate, argument, and modifier must be accented, with the exception of a predicate that, discounting unfocused constituents, is adjacent to an argument.

The first part of SAAR requires every focused constituent to be accented. The second part makes an exception for predicates that meet the specified condition. From a practical point of view it is more convenient to apply SAAR in two steps (Marsi 2001): (1) accent all focused constituents; (2) deaccent the head of the verbal domain if a focused argument is adjacent, discounting unfocused constituents.[10]

The output of this step consists of a string of words marked with accents. The results for each example given in section 3.2.3 are presented in (14)-(17).

In (14a–b) and (16b–c), the verb is accented, because there is no adjacent focused argument.

In (15a) the head of the verbal domain *komt* is deaccented, because it is adjacent to the focused argument NP *de bal*.

In (15b) the head of the verbal domain *gaat* is deaccented, because it is adjacent to the focused argument NP *de rekening*.

In (16a) the modifier *ook* intervenes between the head of the verbal domain *zit* en the argument NP *zo'n vragenlijst*. Since this modifier is unfocused it is ignored, and

[10]Marsi discusses some exceptions for deaccenting the predicate if the adjacent focused argument is topicalised or extraposed. Discussing these issues is beyond the scope of this paper.

zit is deaccented by virtue of its focused argument.

Both verbs in (17a–b) are not accented, because they are the verbal part of a particle verb, which is never accented according to DRV.

(14) a. ik zo om me HEEN HOOR
b. ze MOET daar naar KEMPEN toe

(15) a. via Suat Kaya komt de BAL VOOR
b. de REKENING gaat een BEETJE naar BENEDEN

(16) a. d'r zit ook zo'n VRAGENLIJST bij
b. dat STAAT hier NATUURLIJK niet IN
c. ik PLAK het er OP

(17) a. dat levert een HOOP BESPARINGEN op
b. op de één of andere MANIER straal je dat UIT

3.3 Discussion

The accentuation algorithm has not been implemented and tested. It might turn out that there will be complications that are not foreseen. I want to make a few remarks on the accentuation algorithm that I encountered during its set up.

There is no Focus Projection Rule that projects the focus on a P to the PP. This means that a PP constituent without an NP complement — such as the PP *er op* in Figure 11 — is not recognised by SAAR as a focused argument that might cause the predicate to be deaccented. Further research is required in order to determine whether an extra Focus Projection Rule, such as (18), is necessary.

(18) If a 'VZ2' word within a PP is focused, then the PP is focused as well.

In Grégoire (2004) it was concluded that a particle is not accented if there is an accented adverb in the sentence that causes the rest of the sentence, as well as the particle, not to be accented. This conclusion is not taken into account in the accentuation algorithm since it is not obvious when an adverb is focused. Some adverbs are semantically richer than other adverbs and might be considered content words. With respect to the accentuation of adverbs Marsi (2001) concludes that "[it] is not is not simply a matter of correctly predicting their focus, but requires a semantically driven accentuation rule in the spirit of SAAR." (233)

It was stated before that the accentuation algorithm applies within each of the six discussed verbal domains. The verbal domain 'SSUB' can be dominated by a 'REL' category label of which the head is a pronominal PP — starting with the R-pronoun *waar* ('where') —, and both 'SSUB' and 'SV1' can be respectively dominated by a 'WHSUB' and 'WHQ' category label which head can be the pronominal PP *waarom* ('why'). An example of a 'REL' category is given in Figure 14.
Strictly speaking this means that the verbal domains the accentuation algorithm should apply to must be extended with the domains 'REL', 'WHSUB' and 'WHQ'. Because

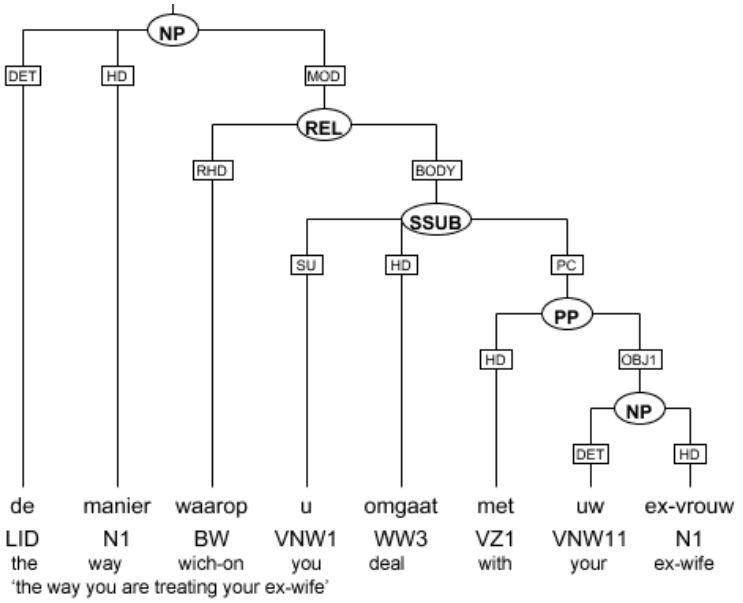


Figure 14: A dependency tree with a 'REL' category.

the pronominal PPs that are the head of these domains are all modifiers — secondary dependency labels are ignored — they are not dealt with by FRA and therefore I stay with the six verbal domains that are distinguished earlier in this chapter and will not add any other domain to them.

Finally it must be noted that the word *naartoe* ('to') that occurs as a stranded preposition is labelled with the PoS tag 'BW'. Although it carries the *d*-label HD within a PP that is an LD, the FRA overlooks it, since it does not have the PoS tag 'VZ2'. This means that either the word *naartoe* should be PoS tagged as a 'VZ2' or the first line of FRA "assign the feature [FOC] to each 'VZ2' word if", should be changed in "assign the feature [FOC] to each 'VZ2' or 'BW' word if".

3.4 Future work

What is left for future work is to evaluate whether the naturalness of synthetic speech improves using the proposed algorithm for accentuation. This can be tested by a listener's experiment. Ideally the algorithm should be implemented in a TTS system in which the PoS tagging and syntactic annotation tasks are performed by the PoS tagger and syntactic annotator used by the CGN project. Since subsequently we need other modules for pre-processing the input text and conversion of the algorithm's output into synthetic speech, this may be too demanding.

A more simple way to test the algorithm is not implement the algorithm exactly as proposed in the previous section, but to integrate the claims formulated in section 2 in

an existing TTS system in which adpositions and particles are never accented.

In the Speech Editor of Fluency (1999, version 1.3),[11] for example, it is possible to add the '\+' or '\-' tag to obtain/delete accent on the word that follows the tag. Without adding any tags, Fluency assigns the accents in a sentence such as *hij belt haar op* ('he called her') as in (19a). According to claim (6c) *op* should be accented, since it is a particle and no focused argument is adjacent. In addition *belt* should be unaccented because I assumed that a particle verb can only carry one accent and if a particle verb is accented the accent goes to the particle and not to the verbal part. Adding the '\+' tag before *op* and the '\-' tag before *belt* results in the accent pattern as in (19b).

(19) a. hij BELT haar op
b. hij belt haar OP

Using this function in Fluency the claims can be tested and subjects can be asked to rate the speech quality of the various sentences.

References

Grégoire, N. (2004). *Accentuation of Adpositions and Particles: Towards a set of rules for predicting accent locations on adpositions and particles for Dutch text-to-speech technology*, Master thesis, University of Utrecht.

Gussenhoven, C. (1992). Sentence accents and argument structure, *in* I.M. Roca (Ed.), *Thematic Structure: its Role in Grammar*, Foris, Berlin, pp. 79–106.

Marsi, E. (2001). *Intonation in Spoken Language Generation*, PhD Dissertation, University of Tilburg.

Marsi, E., Busser, B., Daelemans, W., Hoste, V., Reynaert, M., and Van den Bosch, A. (2002). Combining information sources for memory-based pitch accent placement, *Proceedings of the International Conference on Spoken Language Processing*, pp. 1273–1276.

Odijk, J. (2003). *Herbruikbare woorden en regels*, Inaugural Lecture, University of Utrecht.

Selkirk, E. (1984). *Phonology and Syntax: the Relation between Sound and Structure.*, MIT Press, Cambridge (Mass.).

Selkirk, E. (1995). Sentence accents and argument structure, *in* J. Goldsmith (Ed.), *The Handbook of Phonological Theory*, Blackwell, Cambridge (Mass.), pp. 550–569.

[11] An online demo version of Fluency can be found at http://www.fluency.nl/

Trailfinder: A Case Study in Extracting Spatial Information

Using Deep Language Processing

*Lars Hellan, Dorothee Beermann, Jon Atle Gulla, *Atle Prange*

NTNU, Trondheim, Norway *Businesscape, Trondheim, Norway

Abstract

The present paper reports on an end-to-end application using a deep processing grammar to extract spatial and temporal information of prepositional and adverbial expressions from running text. The extraction process is based on the full understanding of the input text. It is represented in a formalism standard for unification-based grammars and with a language-independent vocabulary as far as spatiotemporal information is concerned. The latter feature in principle allows portability of the extraction algorithm across languages and applications, as long as the domain is kept constant.

The present application is called 'Trailfinder', and supports web-queries about information concerning mountain hikes. A standard hike-description is parsed by an HPSG-based grammar augmented by Minimal Recursion Semantics ('MRS'; (Copestake 2002)). To represent domain-specific meaning concerning location and direction, we enrich MRS structures with feature-based interlingua specifications. Utilizing the 'Heart of Gold' (HoG)[1] technology developed as part of the Deep Thought project[2], and conversion algorithms employing XML sheets, these specifications are mapped to the query interface language.

1 Introduction

One of the problems in arriving at a theoretically satisfactory semantic account of prepositions is their well known polysemy. The Reader's Digest Great Encyclopedic Dictionary for example lists 13 different meanings for the word *behind* - 5 for the adverbial and 8 for the prepositional uses. Many of the most frequent prepositions are in addition ambiguous between a directional and a locative reading, as for example in *The dog runs in the garden*. The directional versus locative interpretation and the adequate semantic modelling of the concepts of PATH and PLACE have been an issue of intensive linguistic research ((Fillmore 1975), (Cresswell 1985), (Talmy 2000), (Jackendoff 1990), and more recently, (Kracht 2002), to mention a few).

In NLP, a reflection of prepositional polysemy is typically encountered in MT, where a single prepositional expression in a source language may correspond to a multitude of expressions in the target language, depending on the object of the prepositional head. The English preposition *on*, e.g., corresponds to the German prepositions *auf, über, an* when combined with an NP expressing place, subject matter or day of the week, respectively. With respect to the problem of determining the correct target language counterpart in such cases, one type of approach which has been developed is symbolic, positing semantic specification in terms of features. Within unification grammar, one of the well known approaches of this type was suggested in (Halvorsen 1995) for Lexical Functional Grammar-based grammar engineering, and

[1] http://heartofgold.dfki.de/

[2] http://www.eurice.de/deepthought

within the machine translation context for prepositional expressions in particular, in (Trujillo 1995), who in turn builds on conceptual distinctions drawn in the linguistic literature (for a survey of this literature, see (Trujillo 1995)).

In the present work we suggest a feature based semantic representation for disambiguation, as part of Minimal Recursion Semantics.

The paper is organized as follows: In section 2 we briefly introduce the Norwegian HPSG grammar 'NorSource', and the Heart of Gold architecture. Section 3 presents the semantics: in section 3.1, we give a short introduction to the MRS formalism, and in section 3.2 we describe our system of sortal specifications on indices as part of the MRS formalism. Section 4 describes the Trailfinder architecture: 4.1 discusses RMRS/XML conversion, and section 4.2 XML transformations. In section 5, we discuss the potential for further developments using the approach instantiated here.

2 The Norwegian HPSG Resource Grammar 'NorSource' and the Heart of Gold

For this project the Norwegian HPSG Resource Grammar 'NorSource'[3] has been used to parse selected sentences from on-line hike descriptions of the Trollheimen region in the middle part of Norway. NorSource was developed as part of the EU-project DeepThought[4] and at present is part of the multilingual grammar engineering initiative Delph-In (http://www.delph-in.net/). It is implemented in the platform Linguistic Knowledge Builder (LKB; Copestake 2002).

In the Trailfinder application, NorSource is used as part of the Heart of Gold component 'PET' ((Callmeier, Schaefer and Siegel 2004)). The Heart of Gold (HoG) is an NLP-architecture which provides, through RMRS ('Robust Minimal Recursion Semantics', cf. (Copestake n.d.)), an interchange format for NLP components of different granularity of processing. In the present application it is used to communicate between parses produced by NorSource and a database for the storage of RMRS representations and a Web Browser. Thus, different from other work on Information Extraction, we do not extract directly from text, but use the markup RMRS produced by our deep processing grammar to encode and store the relevant information. (R)MRS will be described in more detail in the following.

3 Minimal Recursion Semantics

Minimal Recursion Semantics (MRS) representations are 'flat' representations of the elementary predications that compositionally represent the meaning connected to individual constructions, and provide the possibility of underspecifying scope (Copestake et al. 2001, Copestake et al. to appear). The specifications provided by Norsource are, for the present application, in a wholesale fashion carried over to the RMRS markups that we provide for Trailfinder. In the following section we give a short introduction to MRS. Although we use Robust MRS (RMRS) to communicate with Trailfinder, we concentrate in our discussion of prepositional semantics on MRS. (One of the main

[3]More information about NorSource see (http://www.ling.hf.ntnu.no/forskning/norsource).

[4]For more information about the DeepThought project see (http://www.project-deepthought.net)

reasons for the development of RMRS is that it can be used as an output format also for shallower NLP applications such as parts-of-speech parsers and chunkers, and thus allows for a common exchange format between applications of different depth of analysis. For the present application, however, we only work with a deep processing grammar, so that this important aspect of the use of RMRS becomes less relevant.).

3.1 Introduction to MRS

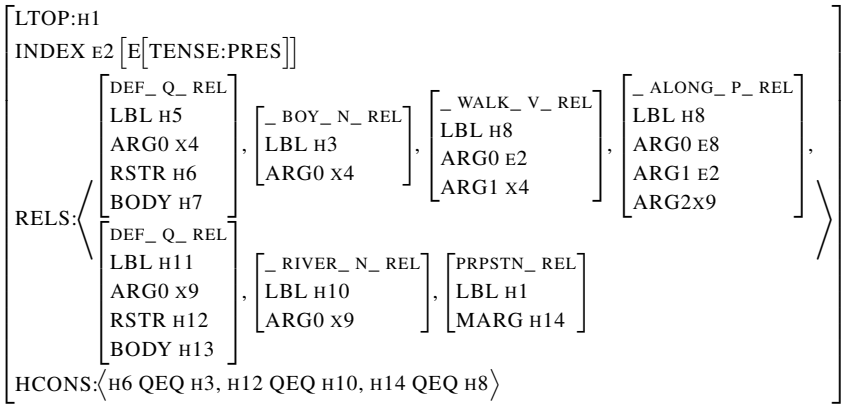


Figure 1: MRS-structure for the sentence *The boy walks along the river*

The above Figure 1 shows a fully specified MRS representation for the sentence

(1) The boy walks along the river

In accordance with a standard MRS set up, for any constituent C (of any rank), the RELS list is a 'bag' of those elementary predications (EPs) that are expressed inside C. The sentence *The boy walks along the river* displays seven EPs, representing the meaning of the six expressions that form this sentence plus one [prpstn_ rel] which reflects the 'message type' of the sentence. The subject argument of the verb *walk* is represented by the coindexation of the verb's ARG1 with the ARG0 of the determiner and the noun that constitute the subject; correspondingly for the ARG2 of the preposition. ARG0 variables are typed: x-type variables correspond to the 'bound variable'of nominal expressions, while 'e' is the type of an event-variable. Scope properties are expressed in the HCONS list, 'x QEQ y' meaning essentially that x scopes over y. HCONS thus records the scopal tree of the constituent in question, as outlined in Copestake et al. (to appear).

The PP *along the river* is interpreted as an event modifier, in the figure represented by the circumstance that the ARG1 of the [_ along_ p_ rel] takes as value the variable 'e2' of the verb, while the handles of the verbal and the prepositional predicate are made identical. A further important feature of MRS structures that carries over to the RMRS structure is that the 'name' of every elementary predication consists of

slots, where the first slot corresponds to the morphological stem, and the second slot indicates its categorial type. This information can be used for further extraction of relevant information; in our case, e.g., we were mainly interested in EPs with the the categorial label _ p.

As mentioned before, the additional semantic sort specifications of the (R)MRS used for Trailfinder are directly provided by NorSource. To this end NorSource was made to process additional sortal class information alongside standard semantic information. In (Hellan and Beermann 2005), we discuss other techniques such as the use of an OWL hierarchy to integrate word sense disambiguation into RMRS. Here we now continue with the presentation of MRS structures that accommodate the additional semantic information.

3.2 Enriched MRS structures

In hiking route descriptions, certain features are prevalent, such as the frequent use of implicit subjects, imperatives and the concatenation of PPs specifying stretches of hikes; the Norwegian text in Figure 5 further below illustrates some of these features. Of further interest to the deep parsing of tour descriptions are verbs of movement in space which in Norwegian are often instantiated as verb-particle constructions, such as *gå-opp/ned/bakover*.[5] Our focus however is on the exploration of prepositional and adverbial senses for the language independent representation of movement in space.

The following are some of the domain specific linguistic features of a text describing hikes: 1. Throughout most of the text, there is a constant 'agent', which can be conceived either as a 'mover', a 'tour', or a road/path - regardless of which perspective is taken, this will be one and the same 'mileage consumer'. An essential aspect of inter-sentential anaphora in these texts is thereby fixed, so that in the summarization of each sentence taken separately, the semantic argument linked to the syntactic subject, that is the ARG1, will have a fixed value. 2. Consecutive sentences, and consecutive directional specifications inside each sentence, generally describe temporally and/or linearly consecutive stretches of path or path-consumption. Also this aspect of intersentential anaphora can be externally superimposed on the representation of each sentence (we return to this issue as we proceed).

An interesting exception are phrases like:

(2) up along the path

where the specifications *up* and *along...*, as long as they are not separated by a comma, typically co-specify the same stretch or move: this situation is represented through the assignment of identical ARG0-values in the EPs representing *up* and *along...*, as opposed to distinct values in the case of consecutive construals. The timeline of a hike is thus reflected in the value of the predicate's ARG0 and distinct ARG0 values map into consecutive 'stages' of the hike.

[5] A further element relevant for information extraction from hike descriptions are place names which often constitute a combination of proper names (e.g., 'Storli') with nouns denoting landscapes, such as 'valley', 'creek', 'lake', etc.; an illustration is given in the translation underneath Figure 5. In the present application we leave place names unanalyzed.

A further analytic concern implemented is the difference between static or locative expressions and directionals. While 'static' modifiers like *in the valley* in phrases like:

(3) they walk around in the valley

(4) the house in the valley

are treated as modifiers predicated of (that is, having as value of their ARG1) the index of the head (that is, for example, *walk* in (3)), directional phrases like:

(5) to the cabin

take as their ARG1 the 'path consumer' expressed, be it as a moving individual or as a road/path. Formally, this is shown by (5) always having an x-variable as its ARG1, whereas ((3) has an e-variable as ARG1. An illustration of the latter case is given in figure 2 below for the preposition *along*. We thus take the approach of (Jackendoff 1990) to directionals and implement the 'mover'as the entity 'measuring out the path'. However, nothing basic to this application hinges on this decision: if we were to treat directionals as event modifiers on a par with stative expressions, both expressions would still be internally distinguished by the value of their SORT attribute. So in short, in the present context, crucial to the summarization of a hiking text is whether a certain location plays the role of starting point, via-point or end point, or of a path or line followed. Important to notice is that in a purely monolingual application, these semantic differences could in principle be accommodated through the representation of the prepositional or adverbial lemmas themselves. However, in a multilingual setting this will not suffice. For example, in an MT application it is the representation of the ambiguity of the English sentence *He walked in the forest* that, for successful generation of a corresponding expression in, e.g., Norwegian or German, needs to lead to two distinct strings, one of which will represent the locative and the other one the directional reading. Likewise for IE, an extraction algorithm based on language independent sortal features is clearly to be preferred over one using language specific features, lending itself more readily to cross linguistic application.

For the application in question, this means that the MRS produced by NorSource will have to supply the arguments of prepositions and adverbs with semantic specifications indicating whether a relation expressed is one of movement to endpoint of path, via viapoint of path, from startpoint of path, or movement along a path: these are, for the time being, specifications under ARG0.SORT.[6] Moreover, for the ARG2 of prepositions (reflecting the governed NP), there will be a SORT specification of whether this is an endpoint, viapoint, etc. This design is illustrated in figure 2 below. The prepositional relation [[_ along_ p_ rel] is annotated with sortal specifications for its ARG0 and its ARG2, indicating that *along* is a preposition of the semantic type 'along-path-motion' and that the ARG2 of this type of preposition denotes a semantic argument of the type 'path-followed'. The full range of prepositions and adverbs in the locative domain are analyzed according to these parameters.

[6]For a development of this approach, see (Hellan and Beermann 2005)

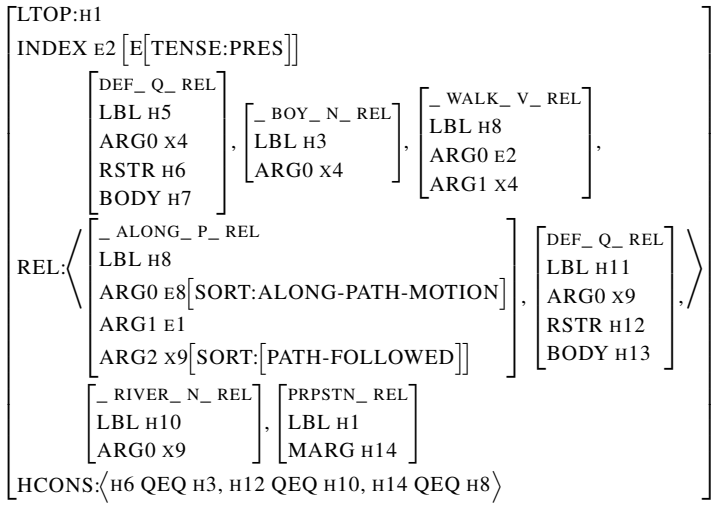


Figure 2: Enriched MRS-structure for the sentence *The boy walks along the river*

4 The Trailfinder Architecture

Trailfinder is a client-server architecture implemented in Java. An administrator regulates the communication with the HoG and cleans and filters the RMRS structures received from the HoG for their final use by the web client. Initially an XML/RPC call is placed to the mocomanserver (HoG). The received RMRS structures are stripped of unnecessary information and stored in the database. In a second step the filtered data is analyzed and placed in a database of tour descriptions accessible to the Search Engine. The Trailfinder architecture is illustrated in figure 3 below. The RMRS received from the HoG is marked up in XML. This makes it relatively easy to filter out the contents of RMRS that are not useful for Trailfinder. The filtering is done with XSLT, and only the nodes marked as EP, ARG0, ARG1 and ARG2 (and their children) are stored in Trailfinder's database. The HoG proxy shown in Figure 3 sends and receives only one sentence at a time. To arrive at a complete tour description, sentences have to be grouped into a single document for storage. Each sentence is considered a stage in a trip. All the senteces are grouped under the name trip in the single XML document. This document is then stored as XML in Trailfinder's database.

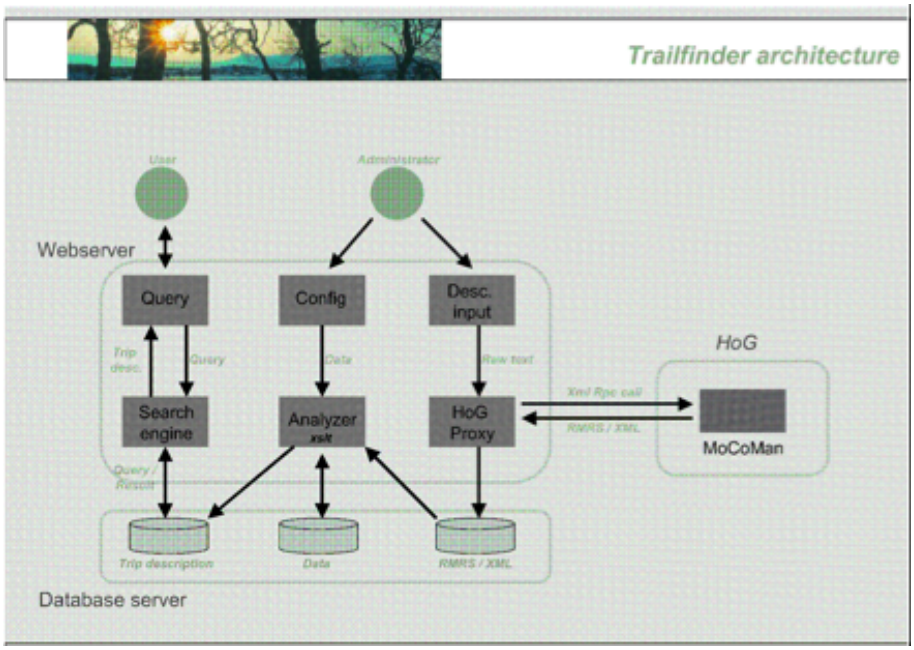


Figure 3: Trailfinder architecture

4.1 Filtering of RMRS and grouping of sentences

The information relevant for Trailfinder can be grouped into three classes: its stages, its vectors and its navigational points. As mentioned above, for the present application, we have made use of the fact that consecutive directional specifications divided by comma inside each sentence generally describe temporally and/or linearly consecutive stretches of path or path-consumption, and furthermore that sentences in general correspond to stages of the hike. Among the occurring exceptions to the latter regularity is the first sentence of the hike description given in Figure 5 further below: instead of describing a stage of the tour, this sentence provides an overall characteristics of the tour (as one that goes 'high and free over the mountain tops'). Still, in our extraction algorithm, we represent also this sentence as a stage of the trip, with *over* as a directional preposition with a 'via-point' sortal specification. To impose the time/path line externally, as we consistently do, such that every sentence corresponds to a partial line on the line that the tour as a whole projects, can thus only serve as a first approximation.

Let us now have a closer look at the information that we filter into the final XML document.

4.2 XML transformation

Figure 4 illustrates the final step of RMRS to the tour description XML. On request by the administrator, the analyzer reads all the RMRS documents and transforms them into a markup that only contains the relevant bits of information for the tour description, the final tour/xml. You find this information listed on the right hand side of Figure 4 below. Next to the 'stages', mentioned in the previous section, we are interested in the vectors a person must follow in order to stay on the tour. This information can be found in the value of the ARG2 of the prepositional relation which corresponds to the variable of the argument NP of the preposition. The variable itself will provide us with further information concerning, e.g. the endpoint of a path, while it is the ARG0 of the prepositional relation itself that provides the sortal specification of the vector as such. Navigational points 'en route', finally, are extracted, e.g., from the string value of CARG (constant argument) of named-relations, from where we extract, e.g., proper names relevant for the tour.

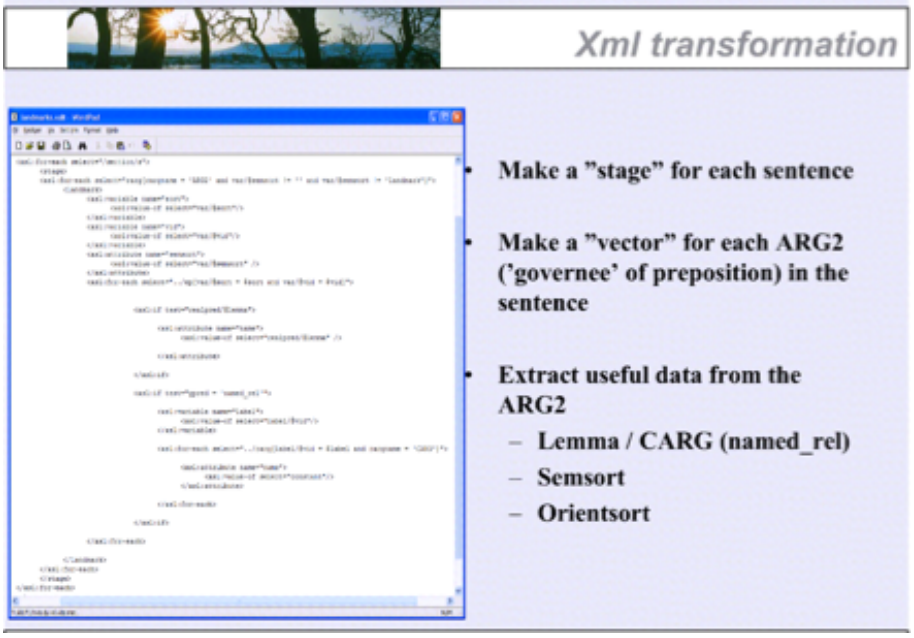


Figure 4: xml-transformation

It should be mentioned at this point that our primary interest is not so much the pictographic summarization format, illustrated on the right hand side of Figure 5 further below, but rather the language independent semantic encoding of basic spatial and temporal relations. This is the topic of the following final section of this paper. Figure 4 summarizes this section, and Figure 5 illustrates an initial sequence of sentences in

its left hand part, and the way they are finally represented in the query interface on the right.

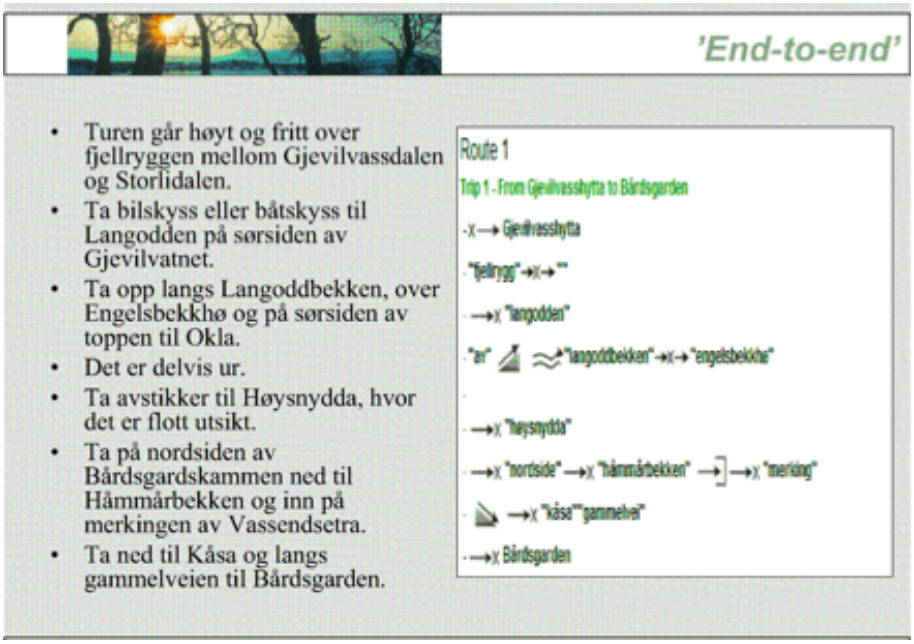


Figure 5: End-to-End

Translation:
(In these translations, brackets supply descriptively relevant parts of the proper names that follow.)
The tour goes high and free over the mountain ridge between [the valley] Gjevilvassdalen and [the valley] Storlidalen. Use car- or boat transportation to Langgodden on the south side of [the lake] Gjevilvannet. Go up along [the creek] Langoddbekken, across [the hill] Engelsbekkshø and on the south side of the top of Okla. The terrain is partly rocky. Take a detour to Høysnydda, from where you have a beautiful view. Go along the north side of the [mountain ridge] Bårdsgårdskammen down to [the creek] Hammarbekken and follow the signs to Vassendsetra. Go down to Kåsa and along the old road to Bårdsgården.

5 A Future for the Trailfinder design

We believe that the more principled interest of the Trailfinder application resides in its utilization of interlingua semantic specifications for spatial and temporal expressions, produced by a deep processing grammar, and the usefulness of this information for

IE purposes. An obvious limitation of the present Trailfinder architecture, is its dependency on the grammar's ability to parse running text. The success of any future application will therefore greatly depend on the future embedding of a grammar such as NorSource into an NLP architecture that combines shallow and deep NLP applications to allow a more adequate coverage of diverse textual input. However, independent of these present limitations in parsing coverage, the main aspect of future interest that emerges from Trailfinder is the circumstance that its subject domain is obviously not restricted to routes in mountains, but that it extends to all textual descriptions of spatial navigation. Following a line of research where feature based lexical semantics is combined with semantic formalism suitable for unification based grammars, two considerations are of special importance: The first one concerns the the outline of a more principled system of conceptual distinctions for the spatiotemporal domain. For the present application we have used a flat list of sortal attributes which are hand-tailored for the present application. In (Hellan and Beermann 2005) we, however, outline a more principled approach to a spatial semantics relevant for the description of prepositional and adverbial meaning. The concept of 'line' and 'x-dimensional' are developed and over 100 spatiotemporal senses, embodied by Norwegian prepositions, are described in what we believe is the beginning of a parsimonious system modelling linguistically relevant spatial concepts.

The second concern for a future extension of the work outlined here is the representation of movement in space. With human-machine communication in mind one possible scenario is to use RMRS-mark-ups to, e.g., inform the movement of artificial agents. For any application that, e.g., relates textual given instructions to robots, success will greatly depend on our ability to model spatial anaphors and also the concept of a time line. The present approach has highlighted some of the issues involved concerning the correlated issue of path-stretches; future work needs to show if, e.g., MRS representation should be used to model temporal sequencing.

References

Callmeier, U.and Eisele, A., Schaefer, U. and Siegel, M.(2004), The deepthought core architecture framework, *in* NN (ed.), *Proceedings of the LREC Conference*, NN, pp. –.

Copestake, A.(2002), *Implementing Typed Feature Structure Grammars*, CSLI Publications, Stanford.

Copestake, A.(n.d.), Report on the design of rmrs, Submitted December 2003.

Cresswell, M.(1985), *Adverbial Modification - Interval Semantics and its Rivals*, D. Reidel, Dordrecht, Holland.

Fillmore, C.(1975), *Santa Cruz lectures on deixis, 1971*, Indiana Unmiversity Linguistics Club.

Halvorsen, P.(1995), *Situation Semantics and Semantic Interpretation in Constraint-Based Grammars*, CSLI Publiscations.

Hellan, L. and Beermann, D.(2005), Classification of prepositional senses for deep grammar applications, *Proceedings from SIGSEM Conference on Prepositions, Univ. of Essex, 2005*, pp. 103–108.

Jackendoff, R.(1990), *Semantic Structures*, MIT Press, Cambridge (Mass).
Kracht, M.(2002), On the Semantics of Locatives, *Linguistics and Philosophy 25*.
Talmy, L.(2000), *Towards a Cognitive Semantics*, MIT Press, Cambridge (Mass).
Trujillo, A.(1995), *Lexicalist Machine Translation of Spatial Prepositions, Ph.D. diss*, Cambridge University.

Learning Dutch Coreference Resolution

Véronique Hoste and Walter Daelemans

CNTS-language Technology Group, University of Antwerp

Abstract

This paper presents a machine learning approach to the resolution of coreferential relations between nominal constituents in Dutch. It is the first significant automatic approach to the resolution of coreferential relations between nominal constituents for this language. The corpus-based strategy was enabled by the annotation of a substantial corpus (ca. 12,500 noun phrases) of Dutch news magazine text with coreferential links for pronominal, proper noun and common noun coreferences. Based on the hypothesis that different types of information sources contribute to a correct resolution of different types of coreferential links, we propose a modular approach in which a separate module is trained per NP type.

1 The task of coreference resolution

Although largely unexplored for Dutch, automatic coreference[1] resolution is a research area which is becoming increasingly popular in natural language processing (NLP) research. It is a weakness and therefore a key task in applications such as machine translation, automatic summarization and information extraction for which text understanding is of crucial importance.

But the resolution of coreferential relations is a complex task since it requires finding the correct antecedent among many possibilities. Furthermore, as shown in example (1) below, it involves different types of knowledge: morphological and lexical knowledge such as number agreement and knowledge about the type of noun phrase, syntactic knowledge such as information about the syntactic function of anaphor and antecedent, semantic knowledge which allows us to recognize synonyms and hyperonyms or which allows distinctions to be made between person, organization or location names, discourse knowledge, world knowledge, etc.

(1) Op 9 november 1983 werd **Alfred Heineken** samen met **zijn** chauffeur ontvoerd. **De kidnappers** vroegen 43 miljoen gulden losgeld. Een bescheiden bedrag, vonden **ze** zelf.
English: On 9 November 1983 **Alfred Heineken** and **his** driver were kidnapped. **The kidnappers** asked a ransom of 43 million guilders. A modest sum, **they** thought.

Whereas corpus-based techniques have become the norm for many other natural language processing tasks (such as part-of-speech tagging, parsing, grapheme-to-phoneme conversion, etc.), the field of computational coreference resolution is still highly knowledge-based, also for Dutch. Among these **knowledge-based approaches**

[1] The discussion whether a given referring link between two nominal constituents can be qualified as coreferential, anaphoric or not is beyond the scope of this paper. We will use both terms interchangeably as is also done in most of the work on computational coreference resolution.

to coreference resolution, a distinction can be made between approaches which generally depend upon linguistic knowledge (Lappin and Leass 1994, Baldwin 1997), and the discourse-oriented approaches, in which discourse structure is taken into account, as in Grosz, Joshi and Weinstein (1995). Beside the fact that not much research has been done yet on automatic coreference resolution for Dutch, the existing research on this topic from op den Akker, Hospers, Lie, Kroezen and Nijholt (2002) and Bouma (2003) falls within the knowledge-based resolution framework and focuses on the resolution of pronominal anaphors. In this paper, we take another perspective and present a machine learning approach to the resolution of coreferential relations between different types of nominal constituents. It is the first corpus-based resolution approach proposed for Dutch. The corpus-based strategy was enabled by the annotation of a new corpus with coreferential relations between noun phrases.

The remainder of this paper is structured as follows. In the following section, we briefly describe the construction and the annotation of the KNACK-2002 corpus. In section 3, we continue with a description of the construction of the data sets. More specifically, we look at the different preprocessing steps that were taken, we consider the construction of positive and negative instances for the training data and test instances for the test data and we motivate the use of three smaller data sets (one for each NP type) instead of one single data set for training and testing. Section 4 gives an overview of the different features that were incorporated in the feature vectors for the machine learning methods we are using. In section 5, we introduce the two machine learning methods which are used for the experiments: memory-based learning and rule-induction. We continue with a description of the experimental setup, viz. the two-step learning approach and the evaluation methodology. Section 6 gives an overview of the experimental results in comparison to two baseline scores. We end this section with a qualitative error analysis of three KNACK-2002 documents. We conclude with a summary.

2 KNACK-2002

Lacking a substantial Dutch corpus of coreferential relations between different types of noun phrases, including named entities, definite and indefinite NPs and pronouns, we annotated a corpus ourselves. This annotation effort was crucial since the existing corpora for Dutch only contain coreferential relations for pronouns and are rather small. The annotated corpus of op den Akker et al. (2002), for example, consists of different types of texts (newspaper articles, magazine articles and fragments from books) and contains 801 annotated pronouns. Another corpus for Dutch was annotated by Bouma (2003). It is based on the Volkskrant newspaper and contains coreferential relations for 222 pronouns.

Our Dutch coreferentially annotated corpus is based on KNACK, a Flemish weekly news magazine with articles on national and international current affairs. KNACK covers a wide variety of topics in economical, political, scientific, cultural and social news. For the construction of this Dutch corpus, we used a selection of articles of different lengths from KNACK, which all appeared in the first ten weeks of 2002.

For the annotation of the Dutch news magazine texts, the following strategy was

taken. First, an annotation scheme was developed containing a set of guidelines for marking up coreferences between noun phrases. On the basis of this annotation scheme, all texts were annotated by two annotators from a pool of five native speakers with a background in linguistics. After the individual coreference annotation by both annotators, they verified all annotations together in order to reach a single consensus annotation rather than keeping several, possibly differing, annotations. In case of no agreement, the relation was not marked. This decision was based on the observations of Hirschman, Robinson, Burger and Vilain (1997) that more than half (56%) of the errors were missing annotations and that 28% of the errors represented "easy" errors (such as the failure to mark headlines or predicating expressions).

The annotation scheme[2] for our Dutch corpus was based on the existing annotation schemes for English. We took the MUC-7 (MUC-7 1998) manual and the manual from Davies, Poesio, Bruneseaux and Romary (1998) as source and we also took into account the critical remarks on these schemes by van Deemter and Kibble (2000). For the annotation of the coreference relations in the KNACK-2002 corpus, we used MITRE's "Alembic Workbench" as annotation environment[3]. The following is an example of such an annotated piece of text:

(2) Ongeveer een maand geleden stuurde <COREF ID = "1"> American Airlines </COREF> <COREF ID = "2" MIN = "toplui"> enkele toplui </COREF> naar Brussel. <COREF ID = "3" TYPE = "IDENT" REF = "1" MIN="vliegtuigmaatschappij"> De grote vliegtuigmaatschappij </COREF> had interesse voor DAT en wou daarover <COREF ID = "5"> de eerste minister </COREF> spreken. Maar <COREF ID = "6" TYPE = "IDENT" REF = "5"> Guy Verhofstadt </COREF> (VLD) weigerde <COREF ID = "7" TYPE = "BOUND" REF = "2"> de delegatie </COREF> te ontvangen.
English: About one month ago, American Airlines sent some senior executives to Brussels. The large airplane company was interested in DAT and wanted to discuss the matter with the prime minister. But Guy Verhofstadt (VLD) refused to see the delegation.

In (2), three coreference chains (sequences of NPs referring to each other) are marked: one for *"American Airlines"* and *"De grote vliegtuigmaatschappij"*, a second chain with *"enkele toplui"* and *"de delegatie"* and a third chain with *"de eerste minister"* and *"Guy Verhofstadt"*. The annotation of this example sentence and all other sentences in our Dutch corpus mainly follows the MUC-7 guidelines (MUC-7 1998). As in the MUC annotations, all coreferences start with a <COREF> tag and are closed with a </COREF> close tag. The initial <COREF> tag contains additional information about the coreference: the unique ID of the NP (ID), the type of coreference relation (TYPE), the ID of the entity referred to (REF) and optionally the minimal tag of the coreference (MIN). For a detailed description of the annotated relations, we refer to Hoste (2005).

In total, the KNACK-2002 corpus consists of 267 documents annotated with coreference information. In this corpus, 12,546 noun phrases are annotated with coreferen-

[2]The annotation scheme is available at http://www.cnts.ua.ac.be/~hoste/proefschrift/AppendixA.pdf.
[3]More information on this workbench can be found at http://www.mitre.org/tech/alembic-workbench.

tial information. Not only did this annotation effort enable us to assess the difficulty of the task, it also led to a corpus which can be used for the evaluation and the development of different approaches to automatic coreference resolution for Dutch.

3 Data preparation

For the experiments, we made a random, but balanced selection of 50 documents covering different topics. We selected 10 documents covering internal politics, 10 documents on foreign affairs, another 10 documents on economy, 5 documents on health and health care, 5 texts covering scientific topics and finally 10 documents covering a variety of topics (such as sports, education, history and ecology). In total, the documents contain 25,994 words and 3,014 coreferential tags. Half of the texts was used as training set and the other half as test set. The division between testing and training material was done randomly at document level (in order to avoid documents being divided in two). The KNACK-2002 training and test set contain 1,688 and 1,326 coreferential NPs, respectively.

3.1 Preprocessing

For the construction of the data sets, we selected all noun phrases in the KNACK-2002 corpus. These noun phrases could be detected after preprocessing the raw text corpora. The following preprocessing steps were taken: tokenization by means of a rule-based system using regular expressions, named entity recognition using the memory-based learning approach of De Meulder and Daelemans (2003), part-of-speech tagging, text chunking and relation finding (all three modules trained on the Spoken Dutch Corpus (CGN)[4] as described in Tjong Kim Sang, Daelemans and Höthker (2004)). We also performed a machine learned morphological analysis (De Pauw, Laureys, Daelemans and Van hamme 2004).

3.2 Instance construction

On the basis of the preprocessed texts, we selected positive and negative instances for the training data and test instances for the test data.

Positive and negative instances As exemplified in Table 1, the positive instances were made by combining each anaphor with each preceding element in the coreference chain. The negative instances were built (i) by combining each anaphor with each preceding NP which was not part of any coreference chain and (ii) by combining each anaphor with each preceding NP which was part of another coreference chain. In order to reduce the number of negative training instances, we restricted the search scope to 20 sentences preceding the candidate anaphor. This instance construction led to a training instance base of 102,376 instances for the 1,687 references in the training data.

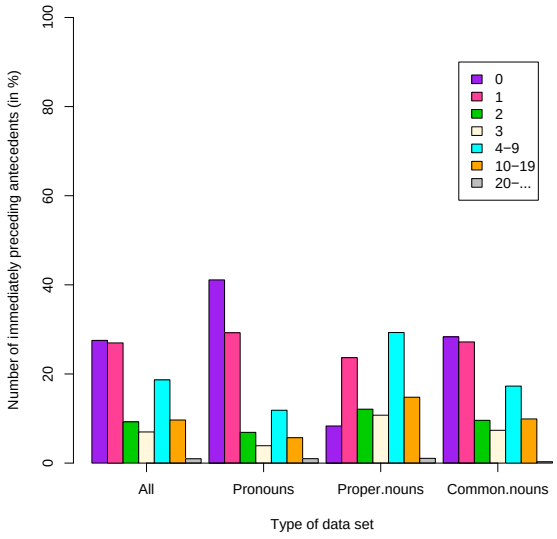
[4] More information on this corpus can be found at http://lands.let.ru.nl/cgn/

Table 1: Training instance construction for the pronoun “ze” as given in example (1).

ze	een bescheiden bedrag	neg
ze	43 miljoen gulden losgeld	neg
ze	de kidnappers	pos
ze	zijn chauffeur	neg
ze	zijn	neg
ze	Alfred Heineken	neg
ze	9 november 1983	neg

Test instances For the construction of the test instances, all NPs starting from the second NP in a text are considered a possible anaphor, whereas all preceding NPs are considered possible antecedents. Since this type of instance construction leads to an enormous increase of the data set and since we are eventually only interested in finding one possible antecedent per anaphor, we took into account some search scope limitations.

Figure 1: Distance in number of sentences between a given referring expression and its immediately preceding antecedent in the KNACK-2002 training set.



As a starting point for restricting the number of instances without losing possibly interesting information, we calculated the distance between the references and their immediately preceding antecedent in the training data. The distances were calculated as follows: antecedents from the same sentence as the anaphor were at distance 0.

Antecedents in the sentence preceding the sentence of the referring expression, were at distance 1, and so on. We divided the group of referring expressions into three categories: (1) pronouns, (2) proper nouns and (3) common nouns. These results are displayed in Figure 1. It shows that for the pronouns 77.3% of the immediately preceding antecedents can be found in a context of three sentences. With respect to the named entities, we can observe that 44.0% of the immediately preceding antecedents can be found in a scope of three sentences. For the common noun NPs, this percentage is 65.2%. We used this information in the construction of the test instances. For the pronouns, all NPs in a context of 2 sentences before the pronominal NP were included in the test sets for the pronouns (as for example also in Yang, Zhou, Su and Tan (2003) for English). For the proper and common nouns, all partially matching NPs were included. For the non matching NPs, the search scope was restricted to two sentences. This instance selection allowed us to obtain an overall test set reduction.

3.3 One vs. three

Instead of merging the different types of NPs into one single training and test set (as for example Ng and Cardie (2002) and Soon, Ng and Lim (2001) for English), we built 3 smaller datasets. This resulted in a learning system for pronouns, one for named entities and a third system for the other NPs. The main motivation for this approach is that other information sources play a role in the resolution of pronominal references than for example in the resolution of references involving proper nouns. Example sentence (3) clearly shows the importance of string matching or aliasing in the resolution of proper nouns. These features are less important for the resolution of the coreferential link between a pronoun and a common noun NP in example (4), for which information on gender, number and distance is crucial.

(3) **Vlaams minister van Mobiliteit Steve Stevaert** dreigt met een regeringscrisis als de federale regering blijft weigeren mee te werken aan het verbeteren van de verkeersveiligheid. (...) **Stevaert** ergert zich aan de manier waarop de verschillende ministeries het dossier naar elkaar toeschuiven.

(4) **De beklaagde**, die de doodstraf riskeert, wil dat **zijn** proces op televisie uitgezonden wordt.

The resulting data sets are displayed in Table 2. The ‘Pronouns’ data set contains the NPs ending on a personal, reflexive or possessive pronoun. The ‘Proper nouns’ data set contains the NPs which have a proper noun as head, whereas the ‘Common nouns’ data set contains all other NPs which are not in the two other categories. And the fourth dataset is the sum of all three datasets. This grouping of the different types of NPs does not only allow for building more specialized classifiers, it also makes error analysis more transparent (as shown in section 7).

4 Selection of informative features

Several information sources contribute to a correct resolution of coreferential relations, viz. morphological, lexical, syntactic, semantic and positional information and

Table 2: Number of instances per NP type in the KNACK-2002 corpus.

	TRAIN		TEST
NP type	positive	negative	
Pronouns	3,111	33,155	5,897
Proper nouns	2,065	31,370	10,954
Common nouns	1,281	31,394	24,677
Complete	6,457	95,919	41,528

also world-knowledge. In this section, we give an overview of the information sources we used for the construction of the instances. These are so-called shallow information sources, namely information sources which are easy to compute.

- The **positional features** give information on the location of the candidate anaphors and antecedents. We use the following three positional features: DIST_SENT (giving information on the number of sentences between the candidate anaphor and its candidate antecedent), DIST_NP (giving information on the number of noun phrases between the candidate anaphor and its candidate antecedent) and the binary feature DIST_LT_THREE (which is set to 'yes' if both constituents are less than three sentences apart from one another and 'no' if both constituents are more than three sentences apart).

- The **local context features** inform on the three words preceding and following the candidate anaphor, with their corresponding part-of-speech tags.

- As **morphological and lexical features**, the I_PRON, J_PRON and I+J_PRON features indicate whether a given candidate anaphor, its candidate antecedent or both are pronouns (personal, possessive, demonstrative or reflexive). The feature J_PRON_I_PROPER indicates whether the possible antecedent of a coreferential pronoun is a proper noun. J_DEMON and J_DEF give information on the demonstrativeness and definiteness of the candidate anaphor. I_PROPER, J_PROPER and BOTH_PROPER indicate whether a given candidate anaphor, its candidate antecedent and both are proper names. And finally, NUM_AGREE looks for number agreement between the candidate anaphor and its candidate antecedent.

- The **syntactic features** ANA_SYNT and ANT_SYNT inform on the syntactic function (subject, object, predicate) of the candidate anaphor and its antecedent. If the candidate antecedent is the immediately preceding subject, object or predicate, it takes as value 'imm_prec_SBJ', 'imm_prec_OBJ' or 'imm_prec_PREDC', respectively. The BOTH_SBJ/OBJ feature checks for syntactic parallelism. The APPOSITIVE feature checks whether the coreferential NP is an apposition to the preceding NP.

- As **string-matching features**, the following features were used: COMP_MATCH, which checks for a complete match between the anaphor

and its candidate antecedent and the PART_MATCH feature, which checks for a partial match between both noun phrases. We also performed word internal matching. In order to do so, we used the previously described morphological analysis to split the compound words into their different parts, e.g. "pensioenspaarverzekeringen" into "pensioen+spaar+verzekeringen" and "pensioenverzekeringen" into "pensioen+verzekeringen'. These different parts were then checked for partial matching. Furthermore, the ALIAS feature indicates whether the candidate anaphor is an alias of its candidate antecedent or vice versa. The alias of a given NP is determined by removing all prepositions and determiners and then by taking the first letter of the nouns in the noun phrase. These letters are then combined in various ways. This simple approach allows us to capture the alias "IBM" which stands for "**I**nternational **B**usiness **M**achines". Finally, the SAME_HEAD feature checks whether the anaphor and its candidate antecedent share the same head. An example of two NPs sharing the same head is "de Golf" and "de Perzische Golf".

- For the extraction of the **semantic features** for the proper nouns , we took into account lists with location names, organization names, person names and male and female person names. Lacking this type of information for the common noun NPs, we used the Celex lexical data base (Baayen, Piepenbrock and van Rijn 1993) instead to provide gender information for the head nouns of the common noun NPs. There are three basic genders in Dutch: male, female and neutral. In addition, CELEX also names female nouns which can be treated as male and nouns whose gender depends on the context in which they are used. This makes five feature values with gender information: 'male', 'female', 'neutral', 'female(male)', 'male-female'. For the extraction of the SYNONYM and HYPERNYM feature, we used all synonyms and hypernyms in the Dutch EuroWordNet (http://www.illc.uva.nl/EuroWordNet) output. And finally, SAME_NE makes use of the output of the Dutch named entity recognition system described earlier.

5 A machine learning approach

5.1 A lazy and an eager learner

Having built the feature vectors for our experiments, we can now continue with a description of the machine learning approaches which we used for our experiments. For the experiments, two machine learning packages were used: the memory-based learning package TIMBL (Daelemans, Zavrel, van der Sloot and van den Bosch 2002)[5], and the rule induction package RIPPER (Cohen 1995). Both TIMBL and RIPPER require as input an example represented as a vector of real-valued or symbolic features, followed by a class. But the two learning methods have a completely different 'bias'. They use different search heuristics and behave differently in the way they represent the learned knowledge.

[5] Available from http://ilk.uvt.nl

The first learning approach applied to our coreferentially annotated data, is a *memory-based learning (MBL)* approach. For our experiments, we used the memory-based learning algorithms implemented in TIMBL (Daelemans et al. 2002). During learning MBL keeps all training data in memory and at classification time, a previously unseen test example is presented to the system and its similarity to all examples in memory is computed using a similarity metric. The class of the most similar example(s) is then used as prediction for the test instance. This strategy is often referred to as "lazy" learning. This storage of all training instances in memory during learning, without abstracting and without eliminating noise or exceptions is the distinguishing feature of memory-based learning in contrast with minimal-description-length-driven or "eager" ML algorithms (e.g. decision trees, rules and decision lists).

The second learning method used in our experiments is the rule learning system RIPPER, which has been developed by Cohen (1995). During learning, RIPPER induces classification rules on the basis of the set of preclassified examples. This type of learning approach is called an eager learning approach, since there is a compression of the training material into a limited number of rules.

5.2 A two-step procedure

The general setup of our experiments is the following. Both RIPPER and TIMBL are trained on the complete training set and the resulting classifiers are applied to the held-out test set, which is represented as a set of instances. Defining the coreference resolution process as a classification problem, however, involves the use of a two-step procedure. In a **first step**, the classifiers are cross-validated on the training data. For this first step, we performed an extensive optimization through feature selection, the optimization of the algorithm parameters and through different sampling techniques in order to have a more balanced class distribution (see Hoste (2005) for a detailed description of this optimization procedure). These optimized classifiers then decide on the basis of the information learned from the training set whether the combination of a given candidate coreference and its candidate antecedent in the test set is classified as a coreferential link. Since each NP in the test set is linked with several preceding NPs, this implies that one single coreference can be linked to more than one antecedent, which for its part can also refer to multiple antecedents, and so on. Therefore, a **second step** is taken, which involves the selection of one coreferential link per coreference. In this second step, the coreferential chains are built on the basis of the positively classified instances. We can illustrate this procedure for the coreferential relation between "hij" and "President Bush" in example (5).

(5) **President Bush** heeft Verhofstadt ontmoet in Brussel. **Hij** heeft met onze eerste minister de situatie in Irak besproken.
English: **President Bush** met Verhofstadt in Brussels. **He** spoke with our prime minister about the situation in Iraq.

For the NP "hij" test instances are built for the NP pairs displayed in Table 3. The result of the first step might be that the learner classifies the first instance as non-coreferential and the last two instances as being coreferential. Since we only want to

Table 3: Test instances built for the "hij" in example (5)

Antecedent	Coreference	Classification
Brussel	hij	no
Verhofstadt	hij	yes
President Bush	hij	yes

select one antecedent per anaphor, a second step is taken to make a choice between the two positive instances (hij - Verhofstadt) and (hij - President Bush). For this **second step**, different directions can be taken. The most straightforward approach is to take as antecedent the first NP found to be coreferent with the anaphor (as in Soon et al. (2001)). Other approaches (Ng and Cardie 2002, Yang et al. 2003) assign scores to the candidate antecedents and select the most likely antecedent among the candidate antecedents. This is also the antecedent selection strategy we have taken (see Hoste (2005) for more information).

5.3 Evaluation procedure

We will report performance in terms of precision, recall and F-measure, using the MUC scoring program from Vilain, Burger, Aberdeen, Connolly and Hirschman (1995). The program looks for the evaluation at equivalence classes, being the transitive closure of a coreference chain. In the Vilain et al. (1995) algorithm, the **recall** for an entire set T of equivalence classes is computed as follows:

$$R_T = \frac{\sum(c(S)-m(S))}{\sum(c(S))}$$

where $c(S)$ is the minimal number of correct links necessary to generate the equivalence class S: $c(S) = (|S| - 1)$. $m(S)$ is the number of missing links in the response relative to equivalence set S generated by the key: $m(S) = (|p(S)| - 1)$. $p(S)$ is a partition of S relative to the response: each subset of S in the partition is formed by intersecting S and the responses sets R_i that overlap S. For the computation of the **precision**, the roles for the answer key and the response are reversed. For example, equivalence class S can consist of the following elements S = {1 2 3 4}. If the response is $< 1 - 2 >$, then $p(S)$ is {1 2}, {3} and {4}.

6 Experimental results

In order to evaluate the performance of our classifiers, we first calculated two baseline scores.

6.1 Two baseline scores

- **Baseline I**: For the calculation of the first baseline, we did not take into account any linguistic, semantic or location information. This implies that this baseline

is calculated on the large test corpus which links every NP to every preceding NP and not on the smaller test corpora described in Section 3 which already take into account feature information. Baseline I is obtained by linking every noun phrase to its immediately preceding noun phrase.

- **Baseline II**: This somewhat more sophisticated baseline is the result of the application of some simple rules: select the closest antecedent with the same gender and number (pronouns), select the closest antecedent which partially/completely matches the NP (proper and common nouns).

Table 4: Two baseline scores. The recall and $F_{\beta=1}$ scores could not be provided for the NP type data sets, since the scoring software does not distinguish between the three NP types.

		Prec.	Rec.	$F_{\beta=1}$
Baseline I	PPC	27.9	**81.9**	41.7
	Pronouns	18.1	—	—
	Proper nouns	2.4	—	—
	Common nouns	4.9	—	—
Baseline II	PPC	**38.9**	45.7	**42.0**
	Pronouns	39.2	—	—
	Proper nouns	56.9	—	—
	Common nouns	23.6	—	—

Table 4 shows the precision, recall and $F_{\beta=1}$ scores for these two baselines. The errors associated with these measures can be interpreted as follows. The recall errors are caused by classifying positive instances as being negative. These false negatives cause missing links in the coreferential chains. The precision errors, on the other hand, are caused by classifying negative instances as being positive. These false positives cause spurious links in the coreferential chains. Table 4 reveals the following tendencies. Linking every NP to the immediately preceding NP, as was done for the first baseline, leads to a high overall recall score of 81.9%, whereas the precision is low: 27.9%. The Baseline II scores which depend on feature information, are more balanced: 45.7% recall and 38.9% precision. The highest $F_{\beta=1}$ value is obtained by Baseline II: 42.0%. With respect to the baseline results on the NP type data sets, the following observations can be made. The Baseline I results are low, except for the precision scores for the pronouns (18.1%). This result confirms that the antecedent of a pronominal anaphor is located close to the anaphor, as already shown in Section 3.

6.2 Classifier results

Table 5 gives an overview of the results obtained by TIMBL and RIPPER in terms of precision, recall and $F_{\beta=1}$. Table 5 shows that both TIMBL and RIPPER obtain an overall $F_{\beta=1}$ score of 51.0%. The precision scores for the "Pronouns" (64.9% for TIMBL and 66.7% for RIPPER) and the "Proper nouns" data sets (79.4% for TIMBL and 79.0% for RIPPER) are much higher than those obtained on the "Common nouns" data set (47.6% for TIMBL and 47.5% for RIPPER). Furthermore, the recall scores

are about 20% lower than the precision scores, which implies that most of the errors represent missing links: 42.2% recall vs. 65.9% precision for TIMBL and 40.9% recall vs. 66.3% precision for RIPPER. Overall, we can conclude from these results that coreference resolution for Dutch still presents some major challenges.

As a test of the methodology used all experiments were also performed on the widely used English MUC-6 and MUC-7 data sets, for which state-of-the art results could be reported: 64.3% (TIMBL) and 63.4% (RIPPER) for MUC-6 and 60.2% (TIMBL) and 57.6% (RIPPER) for MUC-7. For an elaborate description of experiments on these data sets, we refer to Soon et al. (2001), Ng and Cardie (2002) and Hoste (2005).

Table 5: Results from TIMBL and RIPPER in terms of precision, recall and $F_{\beta=1}$. No recall and $F_{\beta=1}$ scores could be provided on the NP type data sets, since the scoring software does not distinguish between the three NP types.

		Prec.	Rec.	$F_{\beta=1}$
Timbl	PPC	65.9	42.2	51.4
	Pronouns	64.9	—	—
	Proper nouns	79.4	—	—
	Common nouns	47.6	—	—
Ripper	PPC	66.3	40.9	50.6
	Pronouns	66.7	—	—
	Proper nouns	79.0	—	—
	Common nouns	47.5	—	—

7 Error analysis

Although we cannot quantify the different types of errors, since this would require a manual analysis of the complete test corpus, we performed a qualitative error analysis on three KNACK-2002 documents. We selected one document on which our system performs above average and two documents for which the $F_{\beta=1}$ score is below average. In each of these documents, we looked for the errors committed by the different learning modules. We will now discuss these errors and some directions for future research.

Pronouns The main source of errors for the pronominal resolution system is the lack of features which can capture the pleonastic and coreferential use of pronouns (as in 7). Therefore, more effort should be put in features which can capture this difference. Another possible approach is to train a classifier, as in Mitkov, Evans and Orasan (2002), which automatically classifies instances of “it” as pleonastic or anaphoric. The resolution of the pronominal anaphors is also hindered by part-of-speech tagging errors (as in 6). E.g. the female “ze” is often erroneously tagged as a third person plural pronoun and vice versa. Furthermore, for the Dutch male and female pronouns, such as “hij”, “hem”, “haar”, the search space of candidate antecedents is much larger

than that for the corresponding English pronouns, since they can also refer to the linguistic gender of their antecedent, as shown in (8).

(6) **De moeder van Moussaoui** gaf een persconferentie waarin **ze** om een eerlijk proces vroeg.
English: **The mother of Moussaoui** gave a press conference in which **she** asked for a fair trial. (Missing link)

(7) Een god van **het vuur**. Paul Wolfowitz heeft alles bij elkaar eigenlijk een bescheiden job in de Amerikaanse regering. Hoe komt **het** dan dat hij zoveel invloed heeft in het Witte Huis?
English: A god of **the fire**. In the end, Paul Wolfowitz has a rather insignificant job in the American government. How is **it** possible that he has so much influence in the White House? (Spurious link)

(8) Zij stelden dat het moeilijk zou zijn om **de studie** te 'dupliceren'. Waarmee werd gezegd dat **ze** niet wetenschappelijk was uitgevoerd.
English: They argued that it would be hard to 'duplicate' **the study**. By which was claimed that **it (Dutch: "she")** was not carried out in a scientific way. (Missing link)

Proper nouns Although high recall and precision scores can be observed for the proper nouns, there is still room for improvement. The errors are mainly caused by preprocessing errors: errors in NP chunking and errors in part of speech tagging (9), etc. The part-of-speech tagger trained on the Spoken Dutch Corpus mainly assigns three different types of tags to proper nouns: SPEC(deeleigen) (as for "Zacarias Moussaoui", SPEC(afgebr) (as for "Moussaoui") and "N(eigen (...)". The corresponding chunks for the underlying part-of-speech tags are "MWU" (multi word unit) for SPEC(deeleigen) and SPEC(afgebr) and "NP" for "N(eigen (...)". Since multi word units can also consist of non-NP combinations (e.g. "in staat"), these multi word units are not always selected for resolution.

(9) **Zacarias Moussaoui**, de eerste persoon die door het Amerikaanse gerecht aangeklaagd is (...) De moeder van **Moussaoui** vloog enige dagen voor zijn voorleiding naar de Verenigde Staten.
English: **Zacarias Moussaoui**, the first person who has been charged by the American judicial authorities (...) The mother of **Moussaoui** came to the United States a few days before the hearing. (Missing link)

Common nouns As also observed for other languages (e.g. Ng and Cardie (2002), Hoste (2005) for English and Strube, Rapp and Müller (2002) for German), the resolution of coreferential relations between common noun NPs is problematic. As for the resolution of coreferential proper nouns and pronouns, missing links can be caused by preprocessing errors, such as errors in part-of-speech tagging, NP chunking, apposition recognition, etc. We therefore conclude that the shallow parser trained on the Spoken Dutch Corpus might not be suitable for this text corpus. We therefore

plan to reconsider the whole preprocessing procedure. Other errors are typical for the resolution of coreferential relations between common nouns: the lack of recognizing synonyms as in (10), the lack of recognizing hyponyms as in (11), and the lack of world knowledge (11). For the construction of the semantic features, we used named entity recognition and the Dutch EuroWordNet. But this lexical resource is very restricted and misses a lot of commonly used expressions and their lexical relations. Furthermore, a lot of coreferential relations are restricted in time, such as the pair "Chirac"-"the president of France", or names of political parties (e.g. "de groenen"-"Agalev"-"Groen!"). In order to overcome this lack of information in the existing resources and in order to capture "dynamic" coreferential relations, we plan to use the Web as a resource (as for example Keller, Lapata and Ourioupina (2002) and Modjeska, Markert and Nissim (2003)).

(10) Stevaert en Charles Picqué gaven elkaar de schuld voor het disfunctioneren van **twee onbemande camera's** op de A12. Picqué - bevoegd voor de erkenning van **de flitspalen** - (...)
English: Stevaert and Charles Picqué blamed each other for the disfunctioning of **two unmanned cameras** at the A12. **Picqué** - authorized for the homologation of **the flash-guns** - (...) (Missing link)

(11) **Zacarias Moussaoui**, de eerste persoon die aangeklaagd is voor **de terreuraanvallen van 11 september**, pleit onschuldig bij zijn eerste verschijning voor de rechtbank. (...) **De Fransman van Marokkaanse afkomst** wordt ervan verdacht de 'twintigste vliegtuigkaper' te zijn die door omstandigheden niet aan **de kapingen** kon deelnemen.
English: Zacarias Moussaoui, the first person who has been charged for **the terrorist attacks of 11 September**, pleads not guilty at the first hearing. (...) **The French citizen of Moroccan descent** is accused of being the 'twentieth hijacker' who was prevented from carrying out **the hijackings**. (Missing link)

8 Summary

In this paper, we presented a machine learning approach to the resolution of coreferential relations between nominal constituents in Dutch. It is the first corpus-based resolution approach proposed for this language. The corpus-based strategy was enabled by the annotation of a corpus with coreferential information for pronominal, proper noun and common noun coreferences: KNACK-2002.

The F scores of 51% of both TIMBL and RIPPER on the held-out test data and the qualitative error analysis showed that coreference resolution for Dutch presents some major challenges. Especially the resolution of coreferential links between common noun NPs is problematic and suffers from lacking semantic and world knowledge. Similar observations could be made for the English MUC-6 and MUC-7 data sets.

References

Baayen, R., Piepenbrock, R. and van Rijn, H.(1993), The celex lexical data base on cd-rom.

Baldwin, B.(1997), Cogniac: high precision coreference with limited knowledge and linguistic resources, *Proceedings of the ACL'97/EACL'97 workshop on Operational Factors in Practical, Robust Anaphora Resolution*, pp. 38–45.

Bouma, G.(2003), Doing dutch pronouns automatically in optimality theory, *Proceedings of the EACL 2003 Workshop on The Computational Treatment of Anaphora*.

Cohen, W. W.(1995), Fast effective rule induction, *Proceedings of the 12th International Conference on Machine Learning (ICML-1995)*, pp. 115–123.

Daelemans, W., Zavrel, J., van der Sloot, K. and van den Bosch, A.(2002), Timbl: Tilburg memory-based learner, version 4.3, reference guide, *Technical Report ILK Technical Report - ILK 02-10*, Tilburg University.

Davies, S., Poesio, M., Bruneseaux, F. and Romary, L.(1998), Annotating coreference in dialogues: Proposal for a scheme for mate, http://www.hcrc.ed.ac.uk/˜ poesio/MATE/anno_manual.htm.

De Meulder, F. and Daelemans, W.(2003), Memory-based named entity recognition using unannotated data, *Proceedings of the Seventh Conference on Natural Language Learning (CoNLL-2003)*, pp. 208–211.

De Pauw, G., Laureys, T., Daelemans, W. and Van hamme, H.(2004), A comparison of two different approaches to morphological analysis of dutch, *Proceedings of the Seventh Meeting of the ACL Special Interest Group in Computational Phonology*, pp. 62–69.

Grosz, B., Joshi, A. and Weinstein, S.(1995), Centering: a framework for modeling the local coherence of discourse, *Computational Linguistics* **21**(2), 203–225.

Hirschman, L., Robinson, P., Burger, J. and Vilain, M.(1997), Automating coreference: The role of annotated training data, *Proceedings of the AAAI Spring Symposium on Applying Machine Learning to Discourse Processing*.

Hoste, V.(2005), *Optimization Issues in Machine Learning of Coreference Resolution*, PhD thesis, Antwerp University.

Keller, F., Lapata, M. and Ourioupina, O.(2002), Using the web to overcome data sparseness, *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP-2002)*, pp. 230–237.

Lappin, S. and Leass, H.(1994), An algorithm for pronominal anaphora resolution, *Computational Linguistics* **20**(4), 535–561.

Mitkov, R., Evans, R. and Orasan, C.(2002), A new, fully automatic version of mitkov's knowledge-poor pronoun resolution method, *Proceedings of the Third International Conference on Intelligent Text Processing and Computational Linguistics (CICLing-2002)*.

Modjeska, N., Markert, K. and Nissim, M.(2003), Using the web in machine learning for other-anaphora resolution, *Proceedings of the 2003 Conference on Empirical Methods in Natural Lanugage Processing (EMNLP-2003)*, pp. 176–183.

MUC-7(1998), Muc-7 coreference task definition. version 3.0., *Proceedings of the Seventh Message Understanding Conference (MUC-7)*.

Ng, V. and Cardie, C.(2002), Combining sample selection and error-driven pruning for machine learning of coreference rules, *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP-2002)*, pp. 55–

62.

op den Akker, H., Hospers, M., Lie, D., Kroezen, E. and Nijholt, A.(2002), A rule-based reference resolution method for dutch discourse, *Proceedings 2002 Symposium on Reference Resolution in Natural Language Processing*, pp. 59–66.

Soon, W., Ng, H. and Lim, D.(2001), A machine learning approach to coreference resolution of noun phrases, *Computational Linguistics* **27**(4), 521–544.

Strube, M., Rapp, S. and Müller, C.(2002), The influence of minimum edit distance on reference resolution, *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP-2002)*, pp. 312–319.

Tjong Kim Sang, E., Daelemans, W. and Höthker, A.(2004), Reduction of dutch sentences for automatic subtitling, *Computational Linguistics in the Netherlands 2003. Selected Papers from the Fourteenth CLIN Meeting*, pp. 109–123.

van Deemter, K. and Kibble, R.(2000), On coreferring: Coreference in muc and related annotation schemes, *Computational Linguistics* **26**(4), 629–637.

Vilain, M., Burger, J., Aberdeen, J., Connolly, D. and Hirschman, L.(1995), A model-theoretic coreference scoring scheme, *Proceedings of the Sixth Message Understanding Conference (MUC-6)*, pp. 45–52.

Yang, X., Zhou, G., Su, S. and Tan, C.(2003), Coreference resolution using competition learning approach, *Proceedings of the 41th Annual Meeting of the Association for Compuatiational Linguistics (ACL-03)*, pp. 176–183.

Shallow Text Analysis and Machine Learning for Authorship Attribution

Kim Luyckx and Walter Daelemans

CNTS-Language Technology Group, University of Antwerp, Belgium

Abstract

Current advances in shallow parsing and machine learning allow us to use results from these fields in a methodology for Authorship Attribution. We report on experiments with a corpus that consists of newspaper articles about national current affairs by different journalists from the Belgian newspaper *De Standaard*. Because the documents are in a similar genre, register, and range of topics, token-based (e.g., sentence length) and lexical features (e.g., vocabulary richness) can be kept roughly constant over the different authors. This allows us to focus on the use of syntax-based features as possible predictors for an author's style, as well as on those token-based features that are predictive to author style more than to topic or register. These style characteristics are not under the author's conscious control and therefore good clues for Authorship Attribution. Machine Learning methods (TiMBL and the WEKA software package) are used to select informative combinations of syntactic, token-based and lexical features and to predict authorship of unseen documents. The combination of these features can be considered an implicit profile that characterizes the style of an author.

1 Introduction

We define Authorship Attribution as the automatic identification of the author of a text on the basis of linguistic features of the text. Applications of Authorship Attribution range from resolving discussions about disputed authorship to forensic linguistics. In this paper, we interpret Authorship Attribution as a text categorization problem. The detection of age, region and gender of the author are other possible applications that could be handled this way, but will not be discussed here.

Automatic Text Categorization (Sebastiani 2002, 2) is a text mining application that labels documents according to a set of predefined content categories. Applications of Text Categorization are numerous. The most important ones are document indexing, document filtering or routing, and the hierarchical categorization of web pages and web search engines. Similar techniques are also being used at sentence level rather than document level for word sense disambiguation. Most Text Categorization systems use a two-stage approach in which (i) automatic feature selection is achieved of features (mostly terms, but also possibly n-grams of terms, NPs, ...) that have high predictive value for the categories to be learned, and (ii) a machine learning approach is used to learn to categorize new documents by using the features selected in the first stage. To allow the selection of linguistic features rather than (n-grams of) terms, robust and accurate text analysis tools such as lemmatizers, part of speech taggers, chunkers etc., are necessary.

An application of this methodology to Authorship Attribution starts from a set of training documents (documents of which the author is known), automatically extracts features that are informative for the identity of the author and trains a machine learn-

ing method that optimally uses these features to do the author attribution for new, previously unseen, documents. Researchers assume that all authors have specific style characteristics that are outside their conscious control. On the basis of those linguistic patterns and markers, the author of a document can be identified (Diederich, Kindermann, Leopold and Paass 2000, 1–2). Rather than designing specific linguistic markers by introspection and testing them by hand, we will use automatic techniques to extract them from text and to test their usefulness in authorship attribution. We will use automatic text analysis tools (a lemmatizer, tagger, and other shallow parser modules) to allow the automatic extraction of potentially relevant linguistic features and patterns.

1.1 Features

We distinguish between four types of features that have traditionally been proposed as being able to differentiate between authors: token-level features (e.g., word length, syllables, *n*-grams), syntax-based features (e.g., part-of-speech tags, rewrite rules), features based on vocabulary richness (e.g., type-token ratio, hapax legomena) and common word frequencies (Stamatatos, Fakotakis and Kokkinakis 2001a). Most studies in the field are based on word forms and their frequencies of occurrence. Studies in the 1950's already were based on token-level features because no powerful computers and robust text analysis software were available (Holmes 1994). But today there are still researchers who use this type of features because it is simple and effective for Authorship Attribution. Stamatatos, Fakotakis and Kokkinakis (2001b) criticize token-level features, although some of their experiments are based on them:

> It is not possible for such measures to lead to reliable results. Therefore, they can only be used as complement to other, more complicated features (Stamatatos et al. 2001b, 195).

Features based on vocabulary richness are more complicated and relevant to an author's style, but have been criticized because they tend to be highly dependent on text length and unstable for texts shorter than 1,000 words (Stamatatos, Fakotakis and Kokkinakis 1999, 162). Common word frequencies can be calculated easily, but selecting the most appropriate words requires some effort.

The contrast between content and function words is basic in Authorship Attribution studies. Authors writing about the same topics tend to use a similar set of content words. Still, those authors have a conscious or unconscious preference for certain other content words. Function words do not seem at first sight to be reliable style markers, since they are very frequent and occur in every text. Nevertheless, the use and frequency of function words is characteristic for authors. An advantage of function words is that they are not under the author's conscious control (Holmes 1994, 90-91). Syntax-based features have been suggested as a different, new, path for capturing style. Though the results are promising (cf. Baayen, Van Halteren and Tweedie (1996), Diederich et al. (2000), Khmelev and Tweedie (2001), Kukushkina, Polikarpov and Khmelev (2001) and Stamatatos et al. (1999)), many researchers try to avoid this type of features because they are hard to compute.

Thanks to improvements in shallow text analysis, we can currently extract reliable syntax-based features. In this paper, we compare token-level, lexical and syntax-based features on a corpus of newspaper articles written by three (groups of) authors. Syntax-based features are extracted by means of the Memory-Based Shallow Parser (MBSP) (Daelemans, Bucholz and Veenstra 1999), which gives an incomplete parse of the input text. MBSP does four types of analysis: it tokenizes the input, performs a Part-of-speech (POS) analysis, looks for noun phrase, verb phrase and prepositional phrase chunks and detects the subject and object of the sentence. The output (of the English MBSP trained on the Wall Street Journal corpus) looks like this:

> [NP1*Subject* POS//NNP tags/NNS NP1*Subject*] [VP1 can/MD be/VB subdivided//VBN VP1] PNP [P into/IN P] [NP open/JJ and/CC NP] PNP [VP2 closed/VBD VP2] [NP2*Object* class/NN words/NNS NP2*Object*] ./.

1.2 Learning Methods

The other focus in Authorship Attribution lies on the classification techniques to be applied. Although there are many techniques for Authorship Attribution, the majority of the studies applies statistical techniques because they are easy to compute and because they are believed to offer an objective method. We will show that the combination of shallow parsing for the automatic construction of predictive features with standard machine learning methods for feature selection and categorization provides an effective methodology for the development of Authorship Attribution systems.

For the Machine Learning experiments, we used a variety of algorithms available in the Waikato Environment for Knowledge Analysis (WEKA)[1] software package (Witten and Frank 1999). We will only report on results obtained with the neural network, traditionally a good approach for working with numeric data, and also one of the WEKA-provided algorithms with which the best results were obtained in exploratory experiments (Luyckx 2004). We also report on experiments using the Tilburg Memory-Based Learner (TiMBL) (Daelemans, Zavrel, van der Sloot and van den Bosch 2004), which is a more advanced $k - nn$-algorithm than the one provided in WEKA. Memory-Based Learning has been proposed as a learning method with the right kind of bias for learning language processing problems because of its ability to learn from untypical or low-frequency events in training data (Daelemans and van den Bosch 2005).

2 Data and Features

The corpus used for training and testing consists of four hundred articles taken from the online archive of the Belgian daily newspaper *De Standaard*[2]. The goal of the experiments was to differentiate between two authors writing about national current affairs. In order to focus on the usefulness of syntactic and token-based features for

[1] WEKA, The University of Waikato: http://www.cs.waikato.ac.nz/~ml/index.html
[2] De Standaard online: http://www.destandaard.be

author rather than topic or register detection, we chose documents within the same range of topics and in the same genre so that there is little difference in vocabulary and register. In order to test the system's robustness, there is a third class of 'other authors'. That way, the system will not only be able to identify the article as written by Anja Otte (class A) or Bart Brinckman (class B) but also as *not* being written by an author from a third class. This O-class consists of articles by ten other authors writing about national current affairs and some collaborative articles by Anja Otte and Bart Brinkman. These may be interesting for later research on the attribution of authorship to articles written by two authors. Table 1 gives an overview of the structure of the training and test corpus.

Author class	Training corpus		Test corpus	
	# articles	# words	# articles	# words
A (Anja Otte)	100 articles	57,682	34 articles	20,739
B (Bart Brinckman)	100 articles	54,479	34 articles	25,684
O (The Others)	100 articles	62,531	32 articles	21,871

Table 1: Training and test corpus

2.1 Features

All features used in this research were automatically extracted using output of the Memory-Based Shallow Parser and the Rainbow system for statistical text classification[3]. We selected nine feature sets of which five are syntax-based. The choice for those specific features is based among others on suggestions made by Glover and Hirst (1995, 4) concerning features based on tagged text. Another feature set (viz. *read*) is based on token information, and we also have two lexical feature sets. Combinations of all features and of all features except the lexical ones are also represented in two separate feature sets. Below is an overview of the feature sets involved in our research:

- *pos*: the frequency distribution of parts-of-speech (POS)
- *verb_B*: the frequency distribution of basic verb forms
- *verb*: the frequency distribution of verb forms
- *pat_num*: the frequency distribution of specific Noun Phrase patterns
- *function*: the frequency distribution of the fourty most frequent function words
- *lex*: the frequency distribution of the twenty most informative words according to the *Rainbow* program
- *read*: the readability score

[3]Rainbow: http://www-2.cs.cmu.edu/ mccallum/bow/rainbow

- *all*: a combination of all features
- *syntax*: a combination of all syntax-based features and the token-level feature *read*

2.1.1 Parts-of-speech

Because most lexical features are highly author and language dependent, rules inferred by Machine Learning classifiers cannot be generalised to other authors or other languages (Stamatatos et al. 1999, 159). Syntax-based features like parts-of-speech do not have this problem because they are not under the conscious control of the author. According to Glover and Hirst (1995, 4), the distribution of parts-of-speech is a possible feature for Authorship Attribution. A list of the POS tags in the feature set and their mean frequency per text in the three author classes can be found below (cf. Table 2):

POS tag	Explanation	Frequency		
		A-class	B-class	O-class
ADJ	adjectives	35	39	41
BW	adverbs	35	30	34
LET	punctuation	79	64	73
LID	articles	59	63	66
N	nouns	121	118	137
SPEC	proper nouns	24	23	20
TSW	interjections	0.3	0.1	0.14
TW	numerals	8	7	14
VG	conjunctions	20	18	25
VNW	pronouns	50	38	48
VZ	prepositions	66	68	78
WW	verbs	81	76	89

Table 2: List of POS tags and their average frequency per text

2.1.2 Verb forms

According to Glover and Hirst (1995, 4), verb forms are also plausible syntax-based style markers. In order to be able to investigate how much grammatical information is needed, we decided to construct separate feature sets for basic and specific verb forms. Kukushkina et al. (2001, 181) found that using detailed information about grammatical classes was less effective than using generalized or 'incomplete' grammatical classes. The basic verb forms used by MBSP are based on the tagset of the Spoken Dutch Corpus (CGN), which distinguishes six verb forms: main verb singular, main verb plural, main verb ending in -t, infinitive, past participle and present participle (Hoekstra, Moortgat, Schuurman and van der Wouden 2001, 84-85). MBSP gives

extra information about these verb forms, so that we end up with seventeen different verb forms.

2.1.3 Noun Phrase patterns

Word-class patterns are syntax-based features that also were proposed in (Glover and Hirst 1995, 4). A first step in investigating whether they are good predictors, is to indicate which specific Noun Phrase patterns occur in our corpus. After that, we construct a document feature vector for the distribution of those patterns. A complex noun phrase like *het sluitstuk van het cipiersakkoord van eind mei* is analysed by MBSP as LID N VZ LID N VZ N N. Most complex noun phrases consist of NP patterns combined by prepositions (VZ) or conjunctions (VG). Therefore, we distinguish twelve frequent NP patterns (cf. Table 3):

Pattern	Example
N, VNW or SPEC	mensen, hij, Albert
ADJ N	snel akkoord
LID N	de regering
N SPEC	voorzitter Verhofstadt
VNW N	zijn partij
TW N	zes maanden
LID ADJ N	de beste kandidaten
N ADJ N	eind vorige week
TW ADJ N	twee overwerkte politici
LID TW N	de vier zwaargewichten
N TW N	zondag 25 december
LID TW ADJ N	een derde nationale steekproef

Table 3: List of np patterns

2.1.4 Function words

The frequency distribution of the fourty most frequent function words in the corpus are represented in the *function* feature set. This allows us to test the relevance of selecting function words as clues for Authorship Attribution.

2.1.5 Content words

This lexical feature set contains binary information about the 20 words with highest mutual information according to the *Rainbow* program for statistical text classification. Mutual Information (MI) is a feature selection method (Sebastiani 2002, 13). We use mutual information to determine which information is shared by the three author classes and which is able to distinguish between them. The 20 words with highest MI selected by Rainbow are *partij*, *SP.A*, *blijkt*, *zegt*, *wie*, *echter*, *altijd*, *aldus*,

evenwel, *blok*, *VLD*, *beide*, *MR*, *gewest*, *tegelijk*, *steeds*, *erg*, *afgelopen*, *momenteel* en *wilde*.

2.1.6 Readability

The readability score is a statistical technique that computes readability based on the average number of syllables per word and the average number of words per sentence (i.e., the Flesch-Kincaid Readability Formula). We want to test whether authors writing for the same newspaper about similar topics have a similar readability[4].

2.1.7 Combination

We also combined the feature sets mentioned above in two separate feature sets: one containing all information and another one only containing token-level (viz. *read*) and syntax-based features. Stamatatos et al. (2001b, 195) state that token-level features alone cannot be useful for Authorship Attribution. They do believe that they can be reliable when used in combination with more complicated features. By combining feature sets, we can test this hypothesis.

3 Machine Learning Approach

Classification of a specific text according to a number of author categories is done by means of Machine Learning. Per document, a feature vector is constructed, containing comma-separated binary or numeric features on the basis of the information described in Section 2.1 and a class label (A, B or O). During training, the Machine Learning algorithms use the information from the training corpus to generate a model by means of which the unseen test instances can be classified. We use the neural network (backprop) implementation of the WEKA software package, and the memory-based classifier TiMBL. In the remainder of this section we briefly discuss and motivate the Machine Learning algorithms we will report the results of.

Artificial Neural Networks consist of a network of units. The input units which represent features are weighted by the strength of their associated connections, and their sum is calculated by a unit receiving input. If the sum is higher than a specified threshold, the output unit fires. The connection weights are computed using the Multi-Layer Perceptron learning rule. Since every author class has its own profile, other sets of features will appear to be meaningful for the A-class than for the B- and C-classes. In our set-up, the neural network has different nodes referring to the three author classes. Classification is performed by checking each test instance against these class thresholds. In our experiments, a backpropagation neural network is used. Training runs through five hundred epochs but is terminated when the error rate increases twenty times in a row. The momentum and the learning rate were fixed at 0.2 and 0.3, respectively.

TiMBL (Memory-based learning) is a supervised inductive algorithm for learning classification tasks based on the $k - nn$ algorithm with various extensions for dealing

[4]Rudolf Flesch: http://www.mang.canterbury.ac.nz/courseinfo/AcademicWriting/Flesch.htm

with nominal features and feature relevance weighting. Memory-based learning stores feature representations of training instances in memory without abstraction and classifies new (test) instances by matching their feature representation to all instances in memory, finding the most similar instances. From these "nearest neighbors", the class of the test item is extrapolated. See Daelemans et al. (2004) for a detailed description of the algorithms and metrics used in our experiments. All memory-based learning experiments were done with the TiMBL software package[5]. In order not to bias the comparison with neural networks (which were used "off the shelf"), we did no extensive model selection (optimization) of the parameters for TiMBL, but we selected the 10 nearest neighbours and added weights using the Information Gain metric.

4 Results

In this Section, we report results with the selected algorithms on the held-out test data. We report on experiments with three (A, B and O) author classes, and compare the use of TiMBL and neural networks for Authorship Attribution.

4.1 Neural Networks

Table 4 gives the results obtained with Neural Networks.

Pos is the best performing syntax-based feature set, with 50.6% F-score. A combination of all syntax-based features increases the F-score (viz., to 61.7%) and has least difficulties identifying the B-class. *Function* outperforms the syntax-based features with 2% in F-score. Combining all features allows the classifier to achieve an F-score of 71.3%, with a highest score on the B-class. For the three-author problem, we see that syntax-based features are able to compete with lexical features but that a combination of syntax-based and lexical features performs best.

Data sets	Author classes			Average
	A-class	B-class	O-class	
pos	34.0%	56.8%	**61.0%**	**50.6%**
verb_B	**45.9%**	43.3%	45.5%	44.9%
verb	**59.3%**	49.1%	41.9%	50.1%
pat_num	48.6%	50.7%	**50.9%**	50.1%
function	**66.7%**	65.7%	59.4%	**63.9%**
lex	54.2%	**71.4%**	53.5%	59.7%
read	**61.1%**	57.5%	25.0%	47.9%
all	70.2%	**74.6%**	69.0%	**71.3%**
syntax	62.1%	**72.3%**	50.8%	**61.7%**

Table 4: Performance on three author classes by Neural Networks in WEKA

[5] Available from http://ilk.uvt.nl

4.2 TiMBL

Table 5 represents results obtained with the memory-based learner TiMBL on three author classes. Considering the F-scores per author does not lead to coherent conclusions. The best syntax-based feature set is *pos*, with 47.7% F-score, while the lexical feature set *function* achieves 54.8%, outperforming the *lex* feature set consisting of content words. Combining all syntax-based features leads to a similar performance (57.3%), while a combination of all features achieves a mean F-score of 72.6%. We see that our syntax-based features achieve better than the *function* lexical feature set. TiMBL performs slightly better on a combination of all features than Neural Networks, but worse on the combination of syntax-based and token-level features.

Data sets	Author classes			Average
	A-class	B-class	O-class	
pos	43.3%	**54.9%**	44.9%	**47.7%**
verb_B	**53.8%**	43.8%	27.6%	41.7%
verb	43.6%	**46.9%**	34.5%	41.7%
pat_num	**53.2%**	50.0%	35.6%	46.3%
function	**65.7%**	55.7%	43.1%	**54.8%**
lex	44.4%	**59.4%**	51.2%	51.7%
read	**62.9%**	53.3%	36.4%	50.9%
all	**77.6%**	74.7%	65.5%	**72.6%**
syntax	59.4%	**61.7%**	50.9 %	**57.3%**

Table 5: Performance on three author classes by Timbl

5 Conclusions

In this paper we proposed a methodology for Authorship Attribution based on the combination of shallow parsing techniques for the extraction of linguistic features with machine learning techniques for feature weighting and author prediction. We illustrated the feasibility of the approach on a corpus consisting of newspaper articles about national current affairs written by three author groups. The linguistic features were computed using the Memory-Based Shallow Parser and Rainbow software packages. We experimented with a multi-class set-up in which the two target authors (categories A and B) had to be identified in a collection of documents in which some documents written by others were present as well (category O).

We compared the performance of a Neural Network (part of the WEKA machine learning software package) and a memory-based learner (TiMBL) for the problem. We found that the three classes can be identified by the Neural Network with an F-score of 71.3% by a combination of token-level, syntax-based and lexical features. Combining syntax-based features leads to an F-score comparable with that of a feature set consisting of the frequency distribution of function words. With a 72.6% F-score, TiMBL does slightly better. Syntax-based features even outperform lexical ones with TiMBL.

Combining syntax-based and token-level features performs almost equally well as or even better than using a lexical feature set. The best syntax-based feature sets are based on the distribution of parts-of-speech. In most cases, the lexical feature consisting of function words works best for the newspaper articles in our corpus. Combining all syntax-based features increases the F-score considerably.

Direct comparison with previous other approaches is impossible, so the following overview of results in related research is of course only indicative. Frequencies of rewrite rules have been shown to be able to distinguish between authors, register and text type in 95% of the documents. Nevertheless, Baayen et al. (1996) point out that their method is too extensive to be used in actual Authorship Attribution practice:

> With the general lack of syntactically annotated text material, it is unlikely that the works in question are available in such an annotated form. (Baayen et al. 1996, 129)

Experiments using frequencies of word forms, word lengths, tagwords and bigrams of tagwords reported on by Diederich et al. (2000) obtained results between 60 and 80 percent. Recall values ranged between 55 and 100 percent for lexical features and between 15 and 40 percent for a combination of token-level and syntax-based features. This shows that our own test results are within line and even considerably better as far as syntax-based features are concerned. Cluster analysis, a statistical technique, can also be applied in Authorship Attribution. On third-person narratives only, frequencies of high-frequency words reach a 87.5% accuracy in work by Hoover (2001, 428). The success rate of Markov chains reaches 83.7% (Khmelev and Tweedie 2001, 306).

Stamatatos et al. (1999) extracted token-level, phrase-level and analysis-level features by means of the Sentence and Chunk Boundaries Detector (SCBD) and found an average error rate of 31% over all authors - and thus a success rate of 69% (Stamatatos et al. 1999, 162). Our combination of lexical, token-level and syntax-based features achieves 72.6% accuracy on the task, which is close to results of the above mentioned studies.

We conclude from our results that the syntax-based, lexical and token-level features we extracted are able to successfully tackle Authorship Attribution problems. As a matter of fact, a combination of our syntax-based features performs almost equally well as and in some cases even better than lexical and token-level features, which were believed to be the most reliable discriminators for authors.

6 Further research

We consider the experiments described here as an explorative study. The results obtained will be compared with methods more common in stylometrics, and the method has to be tested on several other types of text. There is a general worry with newspapers that the texts of the authors are often changed by editor(s).[6]

However, we believe the results clearly open up new perspectives for further research on combining automatically extracted syntax-based features and Machine

[6]Many thanks to the anonymous CLIN reviewer for these and other useful remarks.

Learning techniques for Authorship Attribution. More research will be done on syntax-based features based on parsed text, e.g. the frequency of clause types, syntactic parallelism and the ratio of main to subordinate clauses (Glover and Hirst 1995, 4). We will also explore the different applications of Authorship Attribution, like plagiarism detection and the detection of gender, region, and other properties of the author. Finally, we are currently also using syntax-based features and Machine Learning in a study on Middle-Dutch sermons in order to extract stylistic characteristics.

References

Baayen, H., Van Halteren, H. and Tweedie, F.(1996), Outside the cave of shadows: Using syntactic annotation to enhance authorship attribution, *Literary and Linguistic Computing* **11**(3), 121–131.

Daelemans, W. and van den Bosch, A.(2005), *Memory-Based Language Processing*, Cambridge, UK: Cambridge University Press.

Daelemans, W., Bucholz, S. and Veenstra, J.(1999), Memory-Based Shallow Parsing, *Proceedings of CoNLL-99*, pp. 53–60.

Daelemans, W., Zavrel, J., van der Sloot, K. and van den Bosch, A.(2004), TiMBL: Tilburg Memory Based Learner, version 5.1, reference guide, *Technical Report ILK Research Group Technical Report Series no. 04-02, 2004*, ILK Research Group, University of Tilburg.

Diederich, J., Kindermann, J., Leopold, E. and Paass, G.(2000), Authorship Attribution with Support Vector Machines, *Applied Intelligence* **19**(1-2), 109–123.

Glover, A. and Hirst, G.(1995), Detecting stylistic inconsistencies in collaborative writing, *Writers at work: Professional writing in the computerized environment*.

Hoekstra, H., Moortgat, M., Schuurman, I. and van der Wouden, T.(2001), Syntactic annotation for the Spoken Dutch Corpus project, *Proceedings of the Twelfth Meeting of Computational Linguistics in the Netherlands*, pp. 73–87.

Holmes, D.(1994), Authorship Attribution, *Computers and the Humanities* **28**(2), 87–106.

Hoover, D.(2001), Statistical stylistics and Authorship Attribution: An empirical investigation, *Literary and Linguistic Computing* **6**(4), 421–444.

Khmelev, D. and Tweedie, F.(2001), Using Markov chains for identification of writers, *Literary and Linguistic Computing* **16**(4), 299–307.

Kukushkina, O., Polikarpov, A. and Khmelev, D.(2001), Using literal and grammatical statistics for Authorship Attribution, *Problemy Peredachi Informatsii* **37**(2), 96–108. Translated as Problems of Information Transmission.

Luyckx, K.(2004), *Syntax-based features and Machine Learning techniques for Authorship Attribution*, Master's thesis, University of Antwerp.

Sebastiani, F.(2002), Machine Learning in automated text categorization, *ACM Computing Surveys* **34**(1), 1–47.

Stamatatos, E., Fakotakis, N. and Kokkinakis, G.(1999), Automatic Authorship Attribution, *Proceedings of EACL 99*.

Stamatatos, E., Fakotakis, N. and Kokkinakis, G.(2001a), Automatic text categorization in terms of genre and author, *Computational Linguistics* **26**(4), 471–495.

Stamatatos, E., Fakotakis, N. and Kokkinakis, G.(2001b), Computer-based Authorship Attribution without lexical measures, *Computers and the Humanities* **35**(2), 193–214.

Witten, I. and Frank, E.(1999), *Data Mining: Practical Machine Learning tools with Java implementations*, San Fransisco: Morgan Kaufmann.

Off-line answer extraction for Dutch QA

Jori Mur

Humanities Computing, University of Groningen

Abstract

In Jijkoun et al. [2004] we showed that off-line answer extraction using syntactic patterns is a successful method for answering English factoid questions. In this paper I will discuss the results of applying this method for Dutch question answering using the CLEF text collection parsed by the Alpino parser, a wide coverage dependency parser for Dutch. We defined syntactic patterns to extract answers to frequently occurring question types, such as function questions (*Who is the president of the US?*), location questions (*Where is Groningen located?*) and inhabitant questions (*How many people live in Amsterdam?*).

1 Introduction

Open domain question answering (QA) is a research area that receives much attention nowadays. A QA system takes as input a natural language question, it analyzes the question and tries to find the correct answer in a large text collection. Several strategies have turned out to be quite successful. However, the main part of studies in this domain is focused on QA for English. It would be interesting to see whether similar results can be achieved for other languages for which less resources and less accurate tools are available.

In order to stimulate the development of non-English QA-systems the Cross Language Evaluation Forum (CLEF) organized a question answering track in 2003 and 2004. The QA Track offered tasks to test monolingual and cross-language question answering systems. Resources have been built for several European languages such as Italian, Spanish, Dutch, French and Finnish.

The University of Groningen is working on a Dutch QA-system. One of the techniques we implemented in our system is called *off-line answer extraction*. This technique has proved to be successful (Fleischman et al. [2003]; Jijkoun et al. [2003]). Before actual questions are known, a corpus is exhaustively searched for potential answers to specific question types (`capital, abbreviation, inhabitants, year of birth, ...`). The answers are extracted from the corpus off-line and stored in tables for quick and easy access.

Jijkoun et al. [2004] compared two methods for extracting answers from an English newspaper text collection. They have shown that using syntactic patterns based on dependency relations can lead to significant improvements in recall over methods that are based on regular expression pattern matching.

We implemented the method based on syntactic pattern matching in our Dutch QA system. We parsed a Dutch text collection consisting of two years of newspaper text from two different newspapers (the NRC Handelsblad and the Algemeen Dagblad) using the Alpino Parser (van Noord et al. [2001]). Subsequently, we defined syntactic patterns to extract answers to predefined question categories.

This paper presents the off-line answer extraction module based on syntactic pat-

terns for our Dutch question answering system. In section 2 we discuss related work on answer extraction. In section 3 we describe the experimental framework of our research and in section 4 we discuss our experiments and results. In section 5 you can find conclusions and ideas for future work.

2 Related work

Several QA systems for English have investigated the use of text patterns. Soubbotin and Soubbotin [2001] present a question answering mechanism which uses predefined surface patterns to extract potential answer phrases. After their system achieved the best performance at the TREC-10 evaluation in 2001 more research teams working in the field of corpus-based QA became interested in this technique.

Ravichandran et al. [2003] have investigated the use of surface text patterns for a Maximum Entropy based QA system. The text patterns were collected automatically in an un-supervised way. They have shown that the patterns help to answer more questions correctly.

Fleischman et al. [2003] were the first to present a strategy in which patterns are used to extract answers off-line, before the questions are asked. They evaluated their system on "Who is ..." questions (e.g. person identification: *Who is the mayor of Boston?* and person definition: *Who is Jennifer Capriati?*) against a state-of-the-art web-based QA system. Results indicated that their system answered 25% more questions correctly when it used the extracted information.

Jijkoun et al. [2004] have also evaluated their system on "Who is ..." questions. In contrast to Fleischman et al. they used dependency relations for the extraction of answers off-line. The results showed a significant improvement in recall over systems based on regular expression pattern matching.

3 Experimental Setting

3.1 The QA system

This section describes our open domain QA-system *Joost*. Figure 1 shows an outline of the system architecture. The right branch represents the off-line strategy. We have called this part of the system 'Qatar'. We use a corpus parsed with the Alpino parser, a wide-coverage dependency parser for Dutch. Section 3.2 gives a more detailed description of the parser and its output. Our QA-system scrolls through the parsed corpus to find dependency relations of answer phrases that match the predefined syntactic patterns. The answers found are extracted and stored in tables. More details on the extraction process are given in section 3.3.

Once the tables are created the question answering process can start. An incoming question is analysed. The keywords from the question are selected and the question type is determined. It depends on the question type which step the system will take next. If there is a table associated with the question type, the next step will be to look up the answer in that particular table using the keywords. Answers found by table look-up are ranked according to their frequency. The highest ranked answer is considered as the term with the highest probability to be correct. Candidate answer

which are incorrect, because the author made a mistake or because they were modified, for example by a year ('Amsterdam had 269,000 inhabitants in 1869), most likely have a low frequency and therefore will end up low in the ranking. If no answer is found or if no table was associated with the question type in the first place, the system will follow the other path in the process. In that case, it will use the keywords to retrieve relevant documents, apply natural language processing techniques to re-rank the documents and select the sentence that has the highest probability to contain the answer. Note that answers returned by the online module are complete sentences containing the exact answer phrase. The answers found with the off-line module are exact answers.

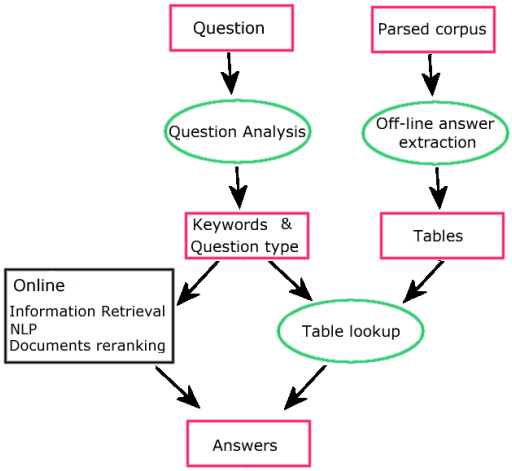


Figure 1: Architecture of our QA-system JOOST

3.2 Alpino

The Alpino system is a linguistically motivated wide-coverage grammar and parser for Dutch. The constraint-based grammar follows the tradition of HPSG Pollard and Sag [1994]. It currently consists of over 500 grammar rules (defined using inheritance) and a large and detailed lexicon (over 100.000 lexemes). To ensure coverage, heuristics have been implemented to deal with unknown words and ungrammatical or out-of-coverage sentences (which may nevertheless contain fragments that are analyzable). The output of the system is a dependency graph, compatible with the annotation guidelines of the Corpus of Spoken Dutch. See an example in figure 2. Malouf and van Noord [2004] show that the accuracy of the system, when evaluated on a test-set of 500 newspaper sentences, is over 88%.

For the QA task the disambiguation model was retrained on a corpus which con-

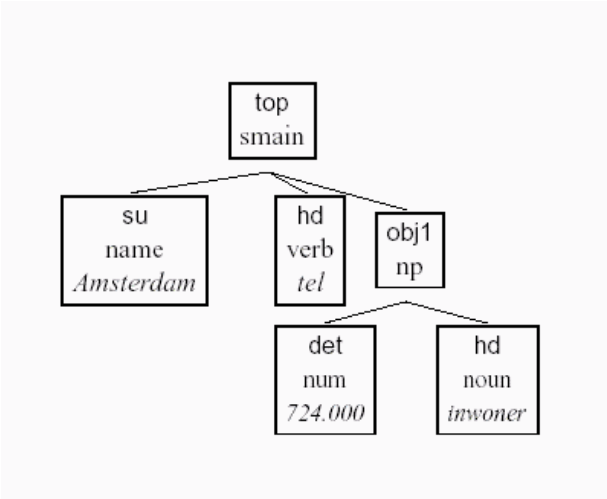


Figure 2: Output for the sentence: Amsterdam telt 724.000 inwoners (*Amsterdam counts 724,000 inhabitants.*)

tained the (manually corrected) dependency trees of 650 quiz questions.[1] The retrained model achieves an accuracy on 92.7% on the CLEF 2003 questions and of 88.3% on CLEF 2004 questions. We have used the Alpino-system to parse the full text collection provided by CLEF for the Dutch QA task.

A second extension of the system for QA, was the inclusion of a Named Entity Classifier. The Alpino system already included heuristics for recognising proper names. Thus, the classifier needed to classify strings which have been assigned a NAME part of speech by grammatical analysis as being of the subtype PER, ORG, GEO or MISC.[2] For unknown names, a maximum entropy classifier was trained, using the Dutch part of the shared task for CONLL 2003.[3] The accuracy on unseen CONLL data of the resulting classifier (which combines dictionary look-up and a maximum entropy classifier) is 88.2%.

3.3 Extraction

We derived patterns based on the structures Alpino produced for potential answers to frequently occurring question types. Table 1 shows how many patterns we created for each question type. The selected question types were in large numbers present in the training set. Answers for the question types listed here tend to occur in more or less

[1] From the *Winkler Prins spel*, a quiz game. The material was made available to us by the publisher, *Het Spectrum, BV.*

[2] Various other entities which sometimes are dealt with by NEC, such as dates and measure phrases, can be identified using the information present in POS tags and dependency labels.

[3] `http://cnts.uia.ac.be/conll2003/ner/`

fixed patterns in the corpus.

Table 1: Question types with # of patterns	
Abbreviations	1
Capitals	2
Currencies	4
Functions	1
Inhabitants	11
Locations	2
Manner of Death	22

For every type an example of a pattern is shown in figure 3 on page 166. The arrows denote dependency relations from head to dependent. Above the arrows are Alpino's dependency labels. *ORG, LOC* en *PER* are tags that are assigned to phrases that are identified by our named-entity tagger as organisations, locations or persons respectively . *X* and *Y* are variables. They can be replaced by every possible word. The phrase *adj* stands for an adjective and *num* for a number.

I will briefly discuss some issues we encountered during the pattern creation process. To find abbreviations we looked for two terms both tagged as 'names' and connected through a modifier relation. Consequently, we extracted much noise as well. The system found the pair 'Tim Burton' and 'Batman', for instance, from the following sentence: '(. . .) op voorwaarde dat Tim Burton (Batman) de regie voor zijn rekening neemt.'(*(. . .) on condition that Tim Burton (Batman) takes care of the production.*) Therefore it was required that all characters from one string also appeared in the same order in the other string.

For the question type 'capital' we extracted the capital and the accompanying country. The country could be formulated as a noun ('Amsterdam, de hoofdstad van **Nederland**' (*Amsterdam, the capital of Holland*)) or as an adjective ('De **Nederlandse** hoofdstad Amsterdam' (*The Dutch capital Amsterdam*)). The same holds for the country name in the currency table. To be able to match both forms with whatever form was found in the question, we used a list of country names along with their adjective forms.

For the function table we defined only one pattern. With this pattern alone we already extracted more than 200,000 facts. Here the problem was that with more patterns we retrieved too much noise. For instance, we wanted a pattern for phrases like 'De Italiaanse minister Agnelli van buitenlandse zaken (*The Italian minister Agnelli for Foreign Affairs*), but the pattern to extract 'Italiaanse' as well as 'van buitenlandse zaken' was too general, that is, it extracted too much noise.

We created many patterns to extract information about inhabitant numbers. It turned out that facts about the number of inhabitants were formulated in numerous different ways. For example the fact that Almere counts 100,000 inhabitants was formulated in the following ways: 'Het 100.000 inwoners tellende Almere', 'Almere (100.000 inwoners)', 'Almere telt 100.000 inwoners', 'Almere met 100.000 inwoners'. The same holds for information about how people died. There are many ways in which people can die.

- **Abbreviation**
  name $\xrightarrow{\text{mod}}$ *name*;
  Centraal Bureau voor de Statistiek (CBS)
  Central Office for Statistics (COS)

- **Capital**
  X $\xleftarrow{\text{mod}}$ `hoofdstad` $\xrightarrow{\text{app}}$ *Y*;
  de Duitse hoofdstad, Berlijn
  the German capital, Berlin

- **Currency**
  adj $\xleftarrow{\text{mod}}$ `munt/munteenheid` $\xrightarrow{\text{app}}$ *noun*;
  de Griekse munteenheid, de Drachme
  the Greek currency, the Drachme

- **Function**
  name(PER) $\xleftarrow{\text{app}}$ *noun* $\xrightarrow{\text{mod}}$ *X*;
  Delors,voorzitter van de Europese Commissie
  Delors, chair of the European Commission

- **Inhabitants**
  name $\xrightarrow{\text{mod}}$ `inwoner` $\xrightarrow{\text{det}}$ *num*;
  Amsterdam (724.000 inwoners)
  Amsterdam (724,000 inhabitants)

- **Location**
  name (ORG/LOC) $\xrightarrow{\text{mod}}$ `in` $\xrightarrow{\text{mod}}$ *name(LOC)*;
  het Rode plein in Moskou
  the Red Square in Moskow

- **Manner of death**
  X $\xleftarrow{\text{subj}}$ `pleeg` $\xrightarrow{\text{obj1}}$ `zelfmoord`;
  Adolf Hitler pleegde zelfmoord
  Adolf Hitler committed suicide

Figure 3: Examples of syntactic patterns

For the location table we searched for information about where organisations and locations where located. However, when we were doing the experiments we discovered that most of the location questions do not ask where something is located, but where something happened ('Waar explodeerde de eerste atoombom?' (*Where did the first atom bomb explode?*)) or it asked for a location with some specific feature ('Welke Europees land heeft de hoogste alcoholconsumptie?'(*Which European country has the highest consumption of alcohol?*)). Due to this fact many location questions were not answered (see section 4, table 5).

The extracted facts were stored as prolog facts. An example fact from the inhabitants table looks as follows:

```
inhabitants('Amsterdam',1,'724.000','AD19950506-0132.xml').
```

The first argument of the predicate `inhabitants` is the location name, the second argument is the number of times this fact was found in the corpus, the third represents the actual number of inhabitants for this location and the last argument gives the ID of the document from which this fact was extracted. The frequency argument and the document ID argument occur in every fact from every table, the other arguments vary depending on the question type.

The abbreviation table contains abbreviations and the associated full terms. In the capitals and currencies tables information is stored about countries and their capitals and currencies respectively. In the function table you can find information about persons and the roles they fulfil in life, for instance: *Bill Clinton, president of America* or *Ralph Fiennes, British actor*. In the location table are organisations, cities, towns and objects stored together with their location. The manner-of-death table contains facts about how people died.

3.4 Corpus and Questions

We extracted the facts from the Dutch CLEF text collection which contains two years of news paper texts from the Algemeen Dagblad and the NRC Handelsblad (1994 and 1995). The total corpus size is 540 MB (200,000 documents) (Magnini et al. [2003b]).

We trained our system on the set of questions provided by CLEF in the DISEQuA corpus (Magnini et al. [2003a]) which were used for the CLEF-2003 QA-track. This question corpus contains 450 questions, each one formulated in four languages, Dutch, Italian, Spanish and English. We only used the Dutch versions.

Some questions were known to have no answer in the text collection. For other questions none of the participating systems found an answer. Those questions were also classified as questions with no answer in the text collection. Since only one group partook in the Dutch monolingual task, a question for which their system did not find an answer was automatically classified as containing no answer. Out of the 450 questions 370 had an answer in the text collection according to this classification. We performed our experiments with the questions that are known to have an answer in the text collection.

We tested our system on the Dutch CLEF-2004 questions. For this year 200 questions were prepared. Out of these 200 questions 185 had an answer in the text collection. Differently from CLEF-2003, where there were only factoid question, twenty

definitional questions ('Wat is het IAEA?' *(What is the IAEA?)*, 'Wie is Alain Juppé?' *(Who is Alain Juppé)*) were included as well. Definition questions tend to be harder to answer then factoid questions, because they expect biographical or descriptive information as an answer rather than a named entity such as a name or a date.

4 Experiments and results

We measured the performance for training and testing by calculating the mean reciprocal rank (MRR), i.e. for each question we take the reciprocal of the rank at which the first correct answer occurs or 0 if none of the returned answers is correct. The score for the whole system is the mean of the individual scores for all the questions. This measure is often used to evaluate QA systems and it is a good score for comparing systems: If a system A ranks the correct answer at the first position for 50 out of 100 questions and at the second position for the other 50 questions, we may assume that this system comes nearer to the perfect system than a system B which also ranks the correct answer at the first position for 50 out of 100 questions, but which does not find an answer for the other 50.

Still, we were also interested in the number of questions for which the correct answer ended up at the first position. Therefore, we also used this as a measure to evaluate the whole system. We evaluated our system using the answers provided by the CLEF organisers.

Table 2: MRR for questions answered by Qatar.

Question set	# Questions	MRR Qatar	MRR Online
CLEF 2003	65/370 (18%)	0.73	0.40
CLEF 2004	24/185 (13%)	0.71	0.46

Table 2 shows how many questions were answered by the off-line module (Qatar), the MRR score for these questions and the MRR for these questions had they been answered by the online module. Remember that with the online module we returned complete sentences, and with Qatar the exact answer, which is harder. For the training set (CLEF 2003) the use of Qatar for these questions results in a very high score (0.73), significantly higher then the score the system would have received if it had used the online approach for answering these questions (0.40). For testing the results were similar: a MRR of 0.71 using Qatar vs. a MRR of 0.46 if the system uses the online module.

The positive effect of the tables is also clear from the results for the whole system on the training data. Table 3 shows the result for the CLEF 2003 data when we take all 370 questions into account. The column labelled - *tables* presents the results the system produced before we implemented Qatar, and the column labelled + *tables* shows the results after adding Qatar to the system.

The MRR increases from 0.49 without tables to 0.54 with the tables. The number of questions for which the correct answer was ranked at the first position increases with almost 30 questions from 145 to 174.

Table 3: Results on CLEF 2003 data.		
	- tables	+ tables
MRR	0.49	0.54
# Correct on 1st	145 (39%)	174 (47%)

Table 4: Results on CLEF 2004 data.		
	- tables	+ tables
MRR	0.51	0.52
# Correct on 1st	77 (42%)	81 (44%)

However, the positive effect was less clear during testing. The results in table 4 on the CLEF 2004 data seem a bit disappointing at first. The MRR increases only 0.01 points from 0.51 without the tables to 0.52 with the tables. In addition, only 4 more questions had the correct answer ranked at the first position.

We take a closer look at the data for testing by looking at each question type separately. The results per question type on the CLEF 2004 data set are shown in table 5. Each row lists the category of the table, the MRR and the number of questions in this category answered by Qatar. For example, three questions were classified as capital questions and all three of them were answered with Qatar. In brackets in the third column are the numbers of questions in the training data for these specific question types. They suggest that the training data was not representative for the test data. In particular the capital questions, the inhabitant questions, the abbreviation questions and location questions appeared not as often in the CLEF 2004 data set as in the CLEF 2003 data set.

Table 5: MRR per question type (CLEF 2004).		
Categories	MRR	# questions
Capital	1.00	3 of 3 (10)
Inhabitants	1.00	1 of 1 (13)
Abbreviation	1.00	1 of 4 (13)
Location	1.00	3 of 20 (62)
Manner of death	0.67	3 of 5 (3)
Function	0.55	11 of 24 (48)
Currency	0.50	2 of 5 (4)

The reason why we see so little effect is because only a small fraction of the questions was answered with Qatar. It is inherent in the setting of CLEF that Qatar answers only a small fraction of the questions, because it is not the intention of the organisers of CLEF to cover exactly the same kind of phenomena as the year before. Therefore some questions will be of an unseen type. Indeed the question set of CLEF 2004 contained question types that did not occur in the CLEF 2003 question set. For example questions that asked for the first name of persons ('Wat is de voornaam van rechter 'Borsellino'?'(*What is judge Borsellino's fist name?*)). We had not created a table for

these kind of questions.

So we may conclude that the trainingdata was not representative for the test data. It would have been better if we had taken part of the CLEF 2003 and part of the CLEF 2004 for training and used the rest for testing. In that case we would also have had a closer resemblance to a QA-system in a real setting. Lowe [2000] claims that Zipf's law applies to user queries. The most frequently asked questions can be classified in only a few question types, in other words, users always tend to ask the same kinds of questions. In that case Qatar would be very suitable.

5 Conclusions and Future work

We described an off-line answer extraction technique which we have implemented in our Dutch QA system Joost. The precision for questions that were answered with our off-line module Qatar was very high. In addition, the performance of the system overall increased. We used the CLEF 2003 data set for training and the CLEF 2004 data set for testing. However, it turned out that the training data was not representative for the test data. That explains why we see so little effect.

In short, the questions that were answered, were almost always correctly answered (high precision), but many questions were not answered (low recall). Therefore, we have to think how to get more questions answered by Qatar. One obvious way to answer more questions using Qatar is to define more patterns for more question types. We could create patterns to extract birth dates or birth places or we could create tables for measures to answer questions about heights and distances (*What is the distance between the sun and the earth?*), for example. Investigating machine learning techniques for finding more patterns would also help extracting more answers and consequently answering more questions. A third option we plan to work on is coreference solution to find more answers. For instance, it is no use to extract that *He* is the *president of the United States*. But if we could find that *He* referred to George Bush (or to Bill Clinton in case of a news paper corpus from 1994 or 1995) we could store that fact in our table.

References

M. Fleischman, E. Hovy, and A. Echihabi. Offline strategies for online question answering: Answering questions before they are asked. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, 2003.

J. Jijkoun, J. Mur, and M. de Rijke. Information extraction for question answering: Improving recall through syntactic patterns. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING)*, 2004.

V. Jijkoun, G. Mishne, and M. de Rijke. Preprocessing documents to answer dutch questions. In *Proceedings of the 15th Belgian-Dutch Conference on Artificial Intelligence (BNAIC 2003)*, 2003.

J.B. Lowe. What's in store for question answering? (invited talk). In *Proceedings of*

the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP/VLC), 2000.

B. Magnini, S. Romagnoli, A. Vallin, J. Herrera, A. Peñas, V. Peinado, F. Verdejo, and M. de Rijke. Creating the disequa corpus: a test set for multilingual question answering. In Peters C., editor, *Working Notes for the CLEF 2003 Workshop*, 2003a.

B. Magnini, S. Romagnoli, A. Vallin, J. Herrera, A. Peñas, V. Peinado, F. Verdejo, and M. de Rijke. The multiple language question answering track at clef 2003. In Peters C., editor, *Working Notes for the CLEF 2003 Workshop*, 2003b.

Malouf and van Noord. Wide coverage parsing with stochastic attribute value grammars. In *IJCNLP-04 Workshop Beyond Shallow Analyses - Formalisms and statistical modeling for deep analyses*, 2004.

C. Pollard and I. Sag. *Head-driven Phrase Structure Grammar*. Center for the Study of Language and Information Stanford, 1994.

D. Ravichandran, A. Ittycheriah, and S. Roukos. Automatic derivation of surface text patterns for a maximum entropy based question answering system. In *Proceedings of the HLT-NAACL Conference*, 2003.

M.M. Soubbotin and S.M. Soubbotin. Patterns of potential answer expressions as clues to the right answer. In *Proceedings of the TREC-10 Conference*, 2001.

G. van Noord, G. Bouma, and R. Malouf. Alpino: Wide-coverage computational analysis of dutch. In *Computational Linguistics in The Netherlands 2000*, 2001.

Syntactic Contexts for Finding Semantically Related Words

Lonneke van der Plas and Gosse Bouma

Humanities Computing, University of Groningen

Abstract

Finding semantically related words is a first step in the direction of automatic ontology building. Guided by the view that similar words occur in similar contexts, we looked at the syntactic context of words to measure their semantic similarity. Words that occur in a direct object relation with the verb *drink*, for instance, have something in common (*liquidity*, ...). Co-occurrence data for common nouns and proper names, for several syntactic relations, was collected from an automatically parsed corpus of 78 million words of newspaper text. We used several vector-based methods to compute the distributional similarity between words. Using Dutch EuroWordNet as evaluation standard, we investigated which vector-based method and which combination of syntactic relations is the strongest predictor of semantic similarity.

1 Introduction

Ontologies comprise semantically related words structured in IS-A relations. An IS-A relation or *hyponym-hypernym* relation holds between a word and a more general word in the same semantic class, e.g. *cat* IS-A *animal*. This type of knowledge is useful for an application such as Question Answering (QA). In many cases, QA systems classify questions as asking for a particular type of *Named Entity*. For instance, given the question *What actor is used as Jar Jar Binks's voice?*, question classification tells the system to look for strings that contain the name of a person. This requires ontological knowledge in which it is stated that an *actor* IS-A *person*. An IS-A hierarchy can also be useful for answering more general WH-questions such as: *What is the profession of Renzo Piano?* In the document collection the following sentence might be found: *Renzo Piano is an architect and an Italian*. Knowing that *Italian* is not a profession but *architect*, helps in deciding which answer to select.

We want to incorporate ontological information in a Dutch QA system. Lexical knowledge bases such as Dutch EuroWordnet (Vossen [1998]) can be used to provide this type of information. However, its coverage is not exhaustive, and thus, we are interested in techniques to automatically extend it. One method to extend an existing IS-A hierarchy is to find words that are semantically related to words already present in the hierarchy. That is, given an ontology which contains an IS-A relation between *banana* and *fruit*, we want to find words related to *banana* (e.g. *orange, strawberry, pineapple, pear, apple, ...*) and include IS-A relations between these words and *fruit* as well.

To find semantically related words, we use a corpus-based method which finds distributionally similar words. Grefenstette [1994] refers to such words as words which have a *second-order affinity*: Words that co-occur frequently (*sinaasappel* (*orange*) and *uitgeperst* (*squeezed*)) have a first-order affinity, words that share the same first-order affinities have a second order affinity, for example, both *sinaasappel* and *citroen* (*lemon*) can be modified by *uitgeperst*.

	hebben (*have*) obj	ziekenhuis (*hospital*) coord	zeggen (*say*) subj	vrouwelijk (*female*) adj	besmettelijk (*contagious*) adj
tandarts	4	4	10	4	0
arts	17	24	148	26	0
ziekte	114	0	0	0	99
telefoon	81	0	0	0	0

Table 1: Sample of the syntactic co-occurrence vectors for various nouns

In this paper, we report on an experiment aimed at finding semantically related words. We briefly discuss previous work on finding distributionally similar words using large corpora. Next, we describe how we collected data for Dutch. Finally, we present the results of an evaluation against Dutch EuroWordNet. We investigated which vector-based methods and which (combinations of) grammatical relations are the strongest predictors of semantic similarity.

2 Related work

2.1 Using Syntactic Context

Words that are distributionally similar are words that share a large number of contexts. There are basically two methods for defining contexts. One can define the context of a word as the n words surrounding it (n-grams, bag-of-words). Another approach is one in which the context of a word is determined by grammatical dependency relations. In this case, the words with which the target word is in a dependency relation form the context of that word.

In both cases, computing distributional similarity requires that a corpus is searched for occurrences of a word, and all relevant words or words plus grammatical relations are counted. The result is a vector. A part of the vectors we collected (using syntactic contexts) for the words *tandarts (dentist), arts (doctor), ziekte (disease)* and *telefoon (telephone)* is given in table 1. Each row represents the vector for the given word. Each column is headed by a word and the grammatical relation it has with the corresponding row word. We can see that *tandarts* appeared four times as the object of the verb *hebben* (*have*) and that *ziekte* never appeared in coordination with *ziekenhuis* (*hospital*).

Kilgarriff and Yallop [2000] use the terms *loose* and *tight* to refer to the different types of semantic similarity that are captured by methods using surrounding words only and methods using syntactic information. The semantic relationship between words generated by approaches which use context only seems to be of a *loose*, associative kind. These methods put words together according to subject fields. For example, the word *doctor* and the word *disease* are linked in an associative way. Methods using syntactic information have the tendency to generate *tighter* thesauri, putting words

together that are in the same semantic class, i.e. words for the *same kind of* things. Such methods would recognise a semantic similarity between *doctor* and *dentist* (both professions, persons, ...), but not between *doctor* and *hospital*. The tighter thesauri generated by methods that take syntactic information into account seem to be more appropriate for ontology building. Therefore, we concentrate on this method.

Most research has been done using a limited number of syntactic relations (Lee [1999], Weeds [2003]). However, Lin [1998a] shows that a system which uses a range of grammatical relations outperforms Hindle's (1990) results that were based on using information from just the subject and object relation. We use several syntactic relations.

2.2 Measures and feature weights

Vector-based methods for finding distributionally similar words, need a way to compare the vectors for any two words, and to express the similarity between them by means of a score. Various methods can be used to compute the distributional similarity between words. Weeds [2003] gives an extensive overview of existing measures. In our experiments, we have only used Cosine and a variant of Dice. These measures are explained in section 3.2. We chose these methods, as they performed best in a large-scale evaluation experiment reported on in Curran and Moens [2002].

The results of vector-based methods can be further improved if we take into account the fact that not all words, or not all combinations of a word and grammatical relation, have the same information value. A large number of nouns can occur as the subject of the verb *hebben* (*have*). The verb *hebben* is selectionally weak (Resnik [1993]) or a *light* verb. A verb such as *uitpersen* (*squeeze*) on the other hand occurs much less frequently, and only with a restricted set of nouns as object. Intuitively, the fact that two nouns both occur as subject of *hebben* tells us less about their semantic similarity than the fact that two nouns both occur as object of *uitpersen*. To account for this intuition, the frequency of occurrence in a vector such as in 1 can be multiplied by a feature weight (each cell in the vector is seen as a feature). The weight is an indication of the amount of information carried by that particular combination of a noun, the grammatical relation, and the word heading the grammatical relation. Various techniques for computing feature weights exist. Curran and Moens [2002] perform experiments using (Pointwise) Mutual Information (MI), the t-test, χ^2, and several other techniques. MI and t-test, the best performing weighting methods according to Curran and Moens, are introduced in section 3.2.

Applying MI to the matrix in 1, results in the matrix in table 2, where frequency counts have been replaced by MI scores. Note that the values for cells involving the verb *hebben* no longer exceed those of the other cells, and that the value for *besmettelijke ziekte* (*contagious disease*) now out-ranks all other values.

2.3 Evaluation

One method for evaluating the performance of a corpus-based method for finding semantically similar words, is to compare the similarity scores assigned by the system

	hebben (*have*) obj	ziekenhuis (*hospital*) coord	zeggen (*say*) subj	vrouwelijk (*female*) adj	besmettelijk (*contagious*) adj
tandarts	0	4.179	0.155	4.158	0
arts	0	3.938	0.540	3.386	0
ziekte	0.550	0	0	0	7.491
telefoon	0.547	0	0	0	0

Table 2: Sample of the MI-weighted syntactic co-occurrence vectors for various nouns

to a pair of words with human judgements. In this form of evaluation, a fixed set of word pairs is used, which are assigned similarity scores by both human judges and the system. If the correlation between the two is high, the system captures human notions of semantic similarity. This evaluation technique has been used for English, using a set of word pairs and human judgements collected originally by Rubenstein and Goodenough [1965]. Resnik [1995] used it to evaluate various measures for computing semantic similarity in WordNet (Fellbaum [1998]) and Weeds [2003] uses it for evaluating distributional measures. Selecting suitable word pairs for comparison, and collecting human judgements for them, is difficult. Furthermore, as Weeds [2003] points out, assigning scores to word pairs is hard for human judges, and human judges tend to differ strongly in the scores they assign to a given word pair.

An alternative evaluation method measures how well similarity scores assigned by the system correlate with similarity in a given lexical resource. Curran and Moens [2002], for instance, computed for each word its nearest neighbours according to a number of similarity measures. Next, they checked whether these pairs were listed as synonyms in one of three different thesauri (the MacQuarie (Bernard [1990]), Moby (Ward [1996]) and Roget (Roget [1911])). A somewhat similar approach is to evaluate nearest neighbours against a lexical resource such as WordNet. A number of measures exist to compute semantic similarity of words in WordNet (Resnik [1995]). A system performs well if the nearest neighbours it finds for a given word are also assigned a high similarity score according to the WordNet measure. An advantage of this evaluation technique is that not only synonyms are taken into account, but also words closely related to the target word. In our experiments, we have used Dutch EuroWordNet (Vossen [1998]) as lexical resource and used the measure of Wu and Palmer [1994]. This method for calculating WordNet similarity is one that correlates well with human judgements according to Lin [1998b] and it can be implemented without the need for frequency information which is difficult to acquire.

3 Experiment

In this section, we describe the data collection process, and the similarity measures and weights we used.

subject-verb	de *kat eet*.
verb-object	ik *voer* de *kat*.
adjective-noun	de *langharige kat* loopt.
coordination	*Bassie en Adriaan* spelen.
apposition	de *clown Bassie* lacht.
prepositional complement	ik *begin met* mijn *werk*.

Table 3: Types of dependency relations extracted

grammatical relation	tuples	types
subject	5.639.140	2.122.107
adjective	3.262.403	1.040.785
object	2642.356	993.913
coordination	965.296	2.465.098
prepositional complement	770.631	389.139
apposition	526.337	602.970

Table 4: Number of tuples and non-identical dependency triples (types) extracted per dependency relation.

3.1 Data collection

As our data we used 78 million words of Dutch newspaper text (Algemeen Dagblad and NRC Handelsblad 1994/1995), that were parsed automatically using the Alpino parser (van der Beek et al. [2002], Malouf and van Noord [2004]). The result of parsing a sentence is a dependency graph according to the guidelines of the Corpus of Spoken Dutch (Moortgat et al. [2000]).

From these dependency graphs, we extracted tuples consisting of the (non-pronominal) head of an NP (either a common noun or a proper name), the dependency relation, and either (1) the head of the dependency relation (for the object, subject, and apposition relation), (2) the head plus a preposition (for NPs occurring inside PPs which are prepositional complements), (3) the head of the dependent (for the adjective and apposition relation) or (4) the head of the other elements of a coordination (for the coordination relation). Examples are given in table 3. The number of tuples and the number of non-identical ⟨`Noun,Relation,OtherWord`⟩ triples (types) found are given in table 4. Note that a single coordination can give rise to various dependency triples, as from a single coordination like *bier, wijn, en noten* (*beer, wine, and nuts*) we extract the triples ⟨*bier, coord, wijn*⟩, ⟨*bier, coord, noten*⟩, ⟨*wijn, coord, bier*⟩, ⟨*wijn, coord, noten*⟩, ⟨*noten, coord, bier*⟩, and ⟨*noten, coord, wijn*⟩. Similarly, from the apposition *premier Kok* we extract both ⟨*premier, hd_app, Kok*⟩ and ⟨*Kok, app, premier*⟩.

For each noun that was found at least 10 times in a given dependency relation

(or combination of dependency relations), we built a vector. Using this cutoff of 10 the matrix built using the the subject relation contained 30.327 nouns, whereas the matrix built using apposition only contained 5.150 nouns. Combining the data for all grammatical relations into a single matrix means that vectors are present for 83.479 nouns.

3.2 Similarity measures used

Methods for computing distributional similarity consist of a measure for assigning weights to the dependency triples present in the matrix, and a measure for computing similarity between two (weighted) word vectors.

As weights we used identity, MI and the t-test. Identity was used as a baseline, and simply assigns every dependency triple a weight of 1 (i.e. every count in the matrix is multiplied by 1).

(Pointwise) Mutual Information (Church and Hanks [1989]) measures the amount of information one variable contains about the other. In this case it measures the relatedness or degree of association between the target word and one of its features. For a word W and a feature f (e.g. the word *ziekte (disease)* and the feature *besmettelijk_adj (contagious_adj)*) is computed as follows:

$$I(W, f) = log \frac{P(W, f)}{P(W)P(f)}$$

Here, $P(W, f)$ is the probability of seeing *besmettelijke ziekte* (in a modifier-head relation) in the corpus, and $P(W)P(f)$ is the product of the probability of seeing *besmettelijke* and the probability of seeing *ziekte*.

An alternative weight method is the t-test. It tells us how probable a certain co-occurrence is. The t-test looks at the difference of the observed and expected mean scaled by the variance of the data. The t-test takes into account the number of co-occurrences of the bi-gram (e.g., a word W and a feature f in a grammatical relation) relative to the frequencies of the words and features by themselves. Curran and Moens [2002] give the following formulation, which we also used in our experiments:[1]

$$t = \frac{P(W, f) - P(W)P(f)}{\sqrt{P(W)P(f)}}$$

We used two different similarity measures to calculate the similarity between two word vectors: *Cosine* and *Dice*† (Curran and Moens [2002]). We describe the functions using an extension of the asterisk notation of Lin [1998b]. An asterisk indicates a set ranging over all existing values of that variable. A subscripted asterisk indicates that the variables are bound together.

Cosine is a geometrical measure. It returns the cosine of the angle between the vectors of the words and is calculated using the dot product of the vectors:

$$Cosine = \frac{\sum_f weight(W1, *_f) \times weight(W2, *_f)}{\sqrt{\sum weight(W1, *)^2 \times \sum weight(W2, *)^2}}$$

[1]Note, however, that this formulation of the t-test differs from that in Manning and Schütze [1999], in spite of the fact that Curran and Moens explicitly refer to Manning and Schütze as their source.

If the two words have the same distribution the angle between the vectors is zero. The maximum value of the *Cosine* measure is 1. *Weight* is either identity, MI or t-test.

Dice is a combinatorial measure that underscores the importance of shared features. It measures the ratio between the size of the intersection of the two feature sets and the sum of the sizes of the individual feature sets. It is defined as:

$$Dice(A,B) = \frac{2. \mid A \cap B \mid}{\mid A \mid + \mid B \mid}$$

,where A stands for the set of features of word 1 and B for the set of features of word 2.

Curran and Moens [2002] propose a variant of Dice, which they call *Dice*†. It is defined as:

$$Dice\dagger = \frac{2\sum_f min(weight(W1,*_f), weight(W2,*_f))}{\sum_f weight(W1,*_f) + weight(W2,*_f)}$$

Whereas *Dice* does not take feature weights into account, *Dice*† does. For each feature two words share, the minimum is taken. If $W1$ occurred 15 times with feature f and $W2$ occurred 10 times with f, and if identity is used for *weight*, it selects 10 as the minimum.

4 Evaluation

Given a matrix consisting of word vectors for nouns, and a similarity method (combination of a weight and similarity measure),the similarity between any pair of nouns can be computed (provided that they are found in the data).[2] On the basis of this, the nouns that are most similar to a given noun can be produced. In this section, we present an evaluation of the system for finding semantically similar words. We evaluated the system against data extracted from EuroWordNet, using various similarity measures and weights, and using various (combinations of) dependency relations.

4.1 Evaluation Framework

The Dutch version of the multilingual resource EuroWordNet (EWN) (Vossen [1998]) was used for evaluation. We randomly selected 1000 target words from Dutch EWN with a frequency of more than 10, according to the frequency information present in Dutch EWN. For each word we collected its 100 most similar words (nearest neighbours) according to our system, and for each pair of words (target word + one of the most similar words) we calculated the semantic similarity according to Dutch EWN. A system scores well if the nearest neighbours found by the system also have a high semantic similarity according to EWN.

EWN is organised in the same way as the well-known English WordNet Fellbaum [1998], that is word senses with the same meaning form *synsets*, and IS-A relations

[2] A demo of the system, using the combination of all grammatical relations, and MI+*Dice*† as similarity method, can be found on `www.let.rug.nl/~gosse/Sets`

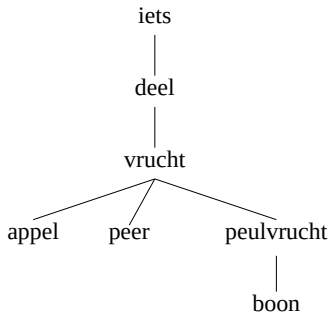


Figure 1: Fragment of the IS-A hierarchy in Dutch EuroWordNet.

between synsets are defined. Together, the IS-A relations form a tree, as illustrated in figure 1. The tree shows that *appel* (*apple*) IS-A *vrucht* (*fruit*), which IS-A *deel* (*part*), which IS-A *iets* (*something*). A *boon* (*bean*) IS-A *peulvrucht* (*seed pod*), which IS-A *vrucht*.

For computing the WordNet similarity between a pair of words we used the Wu/Palmer [1994] measure. It correlates well with human judgements and can be computed without using frequency infomation. The Wu/Palmer measure for computing the semantic similarity between two words W1 and W2 in a wordnet, whose most-specific common ancestor is W3, is defined as follows:

$$Sim = \frac{2(D3)}{D1 + D2 + 2(D3)}$$

We computed, D1 (D2) as the distance from W1 (W2) to the lowest common ancestor of W1 and W2, W3. D3 is the distance of that ancestor to the root node. The similarity between *appel* and *peer* according to the example in 1 would be $4/6 = 0.66$, whereas the similarity between *appel* and *boon* would be $4/7 = 0.57$.

Below, we report EWN similarity for the 1, 5, 10, 20, 50, and 100 most similar words of a given target word. If a word is ambiguous according to EWN (i.e. is a member of several synsets), the highest similarity score is used. The EWN similarity of a set of word pairs is defined as the average of the similarity between the pairs.

4.2 Results

In a first experiment, we compared the performance of the various combinations of weight measures (identity, MI, and t-test) and the measures for computing the distance between word vectors (Cosine and Dice†). The results are given in table 5. All combinations significantly outperform the random baseline (i.e. the score obtained by picking 100 random words as nearest neighbours of a given target word), which, for EWN, is 0.26. Note also that the maximal score is not 1.00, but significantly lower, as words do not have 100 synonyms (which would give, the hypothetical, maximal score

Measure +Weight	EWN Similarity at k=1	k=5	k=10	k=20	k=50	k=100
Dice† +MI	**0.560**	**0.499**	**0.477**	**0.458**	**0.433**	**0.415**
Cosine+MI	0.544	0.489	0.468	0.453	0.428	0.410
Dice† +t-test	0.518	0.482	0.461	0.449	0.425	0.408
Dice† +identity	0.492	0.452	0.430	0.415	0.394	0.375
Cosine+identity	0.494	0.434	0.412	0.396	0.376	0.362
Cosine+t-test	0.472	0.425	0.410	0.402	0.388	0.376

Table 5: Average EWN similarity at k candidates for different similarity measures and weights, using data from the object relation

Dependency Relation	EWN Similarity at k=1	k=5	k=10	k=20	k=50	k=100
Object	**0.560**	**0.499**	**0.477**	**0.458**	**0.433**	**0.415**
Adjective	0.556	0.492	0.463	0.444	0.414	0.395
Coordination	0.495	0.488	0.468	0.453	0.432	0.414
Apposition	0.508	0.465	0.449	0.437	0.418	0.400
Prep. comp.	0.482	0.443	0.431	0.415	0.393	0.380
Subject	0.451	0.426	0.414	0.396	0.380	0.369

Table 6: Average EWN similarity at k candidates for different dependency relations based on Dice† + MI

of 1.00). *Dice*† in combination with MI gives the best results at all points of evaluation, followed by *Cosine* in combination with MI. It is clear that MI makes an important contribution. Also, the difference in performance between *Cosine* and *Dice*† is much bigger when no weight is used (identity) and biggest when t-test is used. t-test and *Cosine* do not work well together, t-test and *Dice*† are a better combination. As *Dice*† *+MI* performs best, this combination was used in the other experiments.

In table 6, the performance of the data collected using various dependency relations is compared. The object relation is best at finding semantically related words. Adjective and coordination are also relatively good, except for the fact that the score for coordination at $k = 1$ is quite a bit lower than for the other two relations. In spite of the fact that using the subject relation most data was collected, this is not a good relation for finding semantically similar words.

In table 7, we give results for various combinations of dependency relations. We started by combining the best performing relations, and then added the remaining relations. In general, it seems to be true that combining data from various relations improves results. Removing the subject relation data from *all*, for instance, decreases performance, in spite of the fact that using only the subject relation leads to poor

	EWN Similarity at					
Combination	k=1	k=5	k=10	k=20	k=50	k=100
Obj	0.560	0.499	0.477	0.458	0.433	0.415
Obj+adj	0.584	0.529	0.499	0.473	0.442	0.420
Obj+adj+coord	0.589	0.533	0.512	0.487	0.459	0.436
Obj+adj+coord+pc	0.585	0.532	0.512	0.491	0.460	0.437
All	**0.603**	**0.542**	0.519	0.494	0.464	0.442
All-appo	0.596	0.541	**0.520**	**0.497**	**0.466**	**0.444**
All-subj	0.588	0.530	0.509	0.488	0.458	0.435

Table 7: Average EWN similarity at k candidates when combining dependency relations based on Dice† + MI

results. The only exception to this rule might be the apposition data. Removing these from *all*, means that slightly better scores are obtained for $k \geq 20$.

4.3 Discussion of results

The fact that MI does so well is at first sight surprising and conflicts with results from earlier research by Curran and Moens [2002]. They show that t-test is the best performing method for setting feature weights. MI in general is known to overemphasise low frequency events. The reason for the fact that MI performs rather well in our experiment could be explained by the cutoffs we set. In section 3.1 we explained that we discarded words that occurred less than 10 times in the relevant configuration.

In accordance with the experiments done by Curran and Moens [2002] we show that *Dice*† outperforms *Cosine*.

From table 6 we can see that there is a difference in performance of the different dependency relations and in table 7 we see that the apposition relation hurts the performance at k =10. However, the evaluation framework is not always a fair one for all relations. Not all similar words found by our system are also found in Dutch EWN. Approximately 60% of the most similar words returned by our system were not found in Dutch EWN. Word pairs found by the system but absent in EWN were discarded during evaluation. This is especially harmful for the apposition relation. The apposition relation always holds between a noun and a proper name. Proper names are not very well presented in EWN, and as a consequence they do not play a role in the evaluation. Therefore, we suspect that the observed effect may well be due to our evaluation method. Other evaluation methods (i.e. in particular a task-based evaluation of using ontological information in QA [3]) may well show that the inclusion of information from appositions has a positive effect. This does suggest that our corpus-based approach indeed finds many words that are absent from the only lexical resource which systematically provides IS-A relations for Dutch, and thus, that automatic or semi-automatic extension of Dutch EWN might be promising.

[3] see van der Plas and Bouma [2005])

dependency relation	Coverage (%)
apposition	11.2
prepositional complement	29.8
object	47.8
adjective	50.3
coordination	56.0
subject	57.9
object+adjective	62.3
object+adjective+coordination	72.9
all-subject-apposition	74.5
all-apposition	78.8
all	78.9

Table 8: Percentage of target words from EWN found in the data set for various (combinations of) dependency relations.

In general we show that combining grammatical relations leads to better results (table 7). In table 8 the percentage of target words that are found in the data collected for different (combinations of) dependency relations (and using a cutoff of 10 occurrences) is given. The fact that coverage increases when combining dependency relations provides further motivation for using systems that combine information from various dependency relations.

The subject relation produces a lot of tuples, but performs surprisingly poorly. Inspection of some sample output, suggests that this may be due to the fact that nouns which denote passive things (i.e. *strawberries* or *tables*) are typically not very well represented in the subject data. Nouns which are clearly agentive, such as *president*, performed much better.

A final note concerns our treatment of coordination. A single coordination consisting of many conjuncts, gives rise to a large number of dependency triples (i.e. the coordination *beer, wine, cheese, and nuts* leads to three dependency triples per word, which is 12 in total). Especially for coordinations involving rare nouns, this has a negative effect. A case in point is the example below, which is a listing of nicknames lovers use for each other:

> Bobbelig Beertje, IJsbeertje, Koalapuppy, Hartebeer, Baloeba Beer, Gerebeer, Bolbuikmannie, Molletje, Knagertje, Lief Draakje, Hummeltje, Zeeuwse Poeperd, Egeltje, Bulletje, Tijger, Woeste Wolf, Springende Spetter, Aap van me, Nunnepun, Trekkie, Bikkel en Nachtegaaltje

This generates 20 triples per name occurring in this coordination alone. As a consequence, the results for a noun such as *aap* (*monkey*) are highly polluted.

5 Conclusion

From our experiment we can conclude that *Dice†* in combination with Mutual Information is the best technique for finding semantically related words. This result is in contrast with results in Curran and Moens [2002].

Another conclusion we can draw is that the object relation is the best performing relation for this task, followed by the adjective relation. The results from coordination can probably be improved, if we adopt a more principled approach to dealing with long coordinations.

However, although some dependency relations perform rather poorly, combining all dependency relations improves the performance of our system. The number of words covered is higher and in almost all cases the average EWN similarity is higher.

In the near future we would like to combine our method for finding similar words with methods for acquiring IS-A relations automatically. Promising results on learning the latter on the basis of data parsed by Alpino are reported in IJzereef [2004]. In addition, we would like to investigate methods for expanding Dutch EWN (semi-)automatically. Finally, we would like to apply the knowledge gathered in this way for QA-tasks, such as question classification, and answering of general WH-questions.

6 Acknowledgements

This research was carried out in the project *Question Answering using Dependency Relations*, which is part of the research program for *Interactive Multimedia Information Extraction*, IMIX, financed by NWO, the Dutch Organisation for Scientific Research.

References

J.R.L. Bernard. The Macquairie encyclopedic thesaurus. The Macquairie Library, Sydney, Australia, 1990.

K.W. Church and P. Hanks. Word association norms, mutual information and lexicography. *Proceedings of the 27th annual conference of the Association of Computational Linguistics*, pages 76–82, 1989.

J.R. Curran and M. Moens. Improvements in automatic thesaurus extraction. In *Proceedings of the Workshop on Unsupervised Lexical Acquisition*, pages 59–67, 2002.

C. Fellbaum. Wordnet, an electronic lexical database. MIT Press, 1998.

G. Grefenstette. Corpus-derived first-, second-, and third-order word affinities. In *Proceedings of Euralex*, pages 279–290, 1994.

D. Hindle. Noun classification from predicate-argument structures. In *Proceedings of ACL-90*, pages 268–275, 1990.

L. IJzereef. Automatische extractie van hyponiemrelaties uit grote tekstcorpora, 2004. URL `www.let.rug.nl/alfa/scripties.html`. Masters thesis, Rijksuniversiteit Groningen.

A. Kilgarriff and C. Yallop. What's in a thesaurus? In *Proceedings of the Second Conference on Language Resource an Evaluation*, pages 1371–1379, 2000.

L. Lee. Measures of distributional similarity. In *37th Annual Meeting of the Association for Computational Linguistics*, pages 25–32, 1999.

Dekang Lin. Automatic retrieval and clustering of similar words. In *COLING-ACL*, pages 768–774, 1998a.

Dekang Lin. An information-theoretic definition of similarity. In *Proc. 15th International Conf. on Machine Learning*, pages 296–304. Morgan Kaufmann, San Francisco, CA, 1998b.

Robert Malouf and Gertjan van Noord. Wide coverage parsing with stochastic attribute value grammars. In *IJCNLP-04 Workshop Beyond Shallow Analyses - Formalisms and stati stical modeling for deep analyses*, Hainan, 2004.

Christopher D. Manning and Hinrich Schütze. *Foundations of Statistical Natural Language Processing*. The MIT Press, Cambridge, Massachusetts, 1999.

Michael Moortgat, Ineke Schuurman, and Ton van der Wouden. CGN syntactische annotatie, 2000. Internal Project Report Corpus Gesproken Nederlands, see http://lands. let.kun.nl/cgn.

P. Resnik. Selection and information. Unpublished doctoral thesis, University of Pennsylvania, 1993.

P. Resnik. Using information content to evaluate semantic similarity. In *Proceedings of the 14th international joint conference on artificial intelligence*, pages 448–453, 1995.

P. Roget. Thesaurus of English words and phrases, 1911.

H. Rubenstein and J.B. Goodenough. Contextual correlates of synonymy. *Communications of the ACM*, 1965.

Leonoor van der Beek, Gosse Bouma, and Gertjan van Noord. Een brede computationele grammatica voor het Nederlands. *Nederlandse Taalkunde*, 7(4):353–374, 2002.

Lonneke van der Plas and Gosse Bouma. Automatic acquisition of lexico-semantic knowledge for QA. Proceedings of Ontolex, 2005. to appear.

P. Vossen. Eurowordnet a multilingual database with lexical semantic networks, 1998. URL `citeseer.ist.psu.edu/vossen98eurowordnet.html`.

G. Ward. Moby thesaurus. Moby Project, 1996.

J. Weeds. *Measures and Applications of Lexical Distributional Similarity*. PhD thesis, University of Sussex, 2003.

Z. Wu and M. Palmer. Verb semantics and lexical selection. In *The 23rd Annual Meeting of the Association for Computational Linguistics*, pages 133–138, 1994.

Extended Lexical Units in Dutch

Michaela Poß and Ton van der Wouden

The Leiden University Centre for Linguistics (LUCL)

Abstract

The paper describes ongoing empirical research into a fundamental problem of linguistics, viz. the architecture of grammar, or the division of labor between lexicon and grammar. We try to find an answer to this question by investigating which part of the utterances in a recent corpus of spontaneous spoken Dutch consists of "Extended Lexical Units" (ELU's), hypothesized to be stored in the lexicon, rather than new syntactic constructs creatively generated from lexical atoms. We describe some problems involved in the identification of these ELUs and in the implementation of them in an NLP system. For the latter, we assess the usability of basic assumptions from frameworks such as Construction Grammar and Head-driven Phrase Structure Grammar.

1 Introduction: The importance of ELUs in Language Use

Recent developments in linguistic theory are starting to put into question the traditional picture of the language system consisting of an interesting grammar *vis à vis* a boring lexicon. Large parts of everyday spoken language are arguably constructed out of "extended lexical units" (ELUs), which we will use as a pretheoretical term to refer to all linguistic building blocks larger than words, be they compositional or not, that must be assumed to be stored in the lexicon (sometimes also known as "constructicon"), because they have idiosyncratic properties as regards their phonology, morphology, syntax, semantics, pragmatics, style level, etc. Note that lexical storage of these ELUs does not preclude the possibility that they possess various degrees of grammatical structure and/or grammatical freedom.

In any case, the very existence of these ELUs raises fundamental questions with respect to the architecture of the grammar faculty (Jackendoff 1997). From another perspective, these ELUs are a key problem for the development of large-scale, linguistically sound natural language processing technology (Sag, Baldwin, Bond, Copestake and Flickinger 2001).

The time pressure inherent to spontaneous speech situations leaves less time for the complex mental computations involved in language production than is the case in the scrupulous composition of written text. One of the strategies for speakers to overcome this problem is to employ ELUs to construct their utterances, instead of being completely original. Kuiper (1996) observes that in certain high-pressure situations, most of speakers' utterances consist of stored (or, to use another metaphor, pre-compiled) linguistic material, with very little syntactic computation going on. We may assume that this is one extreme of a cline, the other extreme being Chomsky's idealization of a creative language user with an infallible memory and infinite processing power, with enough time to verbalize new ideas in an original way.

Time pressure is not the only reason for extensive usage of ELUs, ritualization of speech situations, or of social interaction in general, being another (Wray 2002).

If a speaker of English wants to convince his or her conversation partner that he/she is following the partner's line of reasoning, he/she can do so by saying such a thing as *I see*, and if one wants to get the attention of a more or less formal gathering of people, one can raise one's voice and chant *ladies and gentlemen*. The last example immediately shows two aspects of the fixedness of this ELU: it (usually) works, even if the speaking person would never even think of individually addressing the members of audience with *lady* or *gentleman*, and the reverse order *gentlemen and ladies* is far less effective, to say the least.[1]

In the literature, estimates concerning the quantitative importance of ELUs differ greatly: Sprenger (2003) reports that some 10% of the content words in a corpus of Dutch newspaper text was part of some larger lexical entity whereas Altenberg (1998) writes: "A rough estimation indicates that over 80 per cent of the words in the corpus form part of a recurrent word-combination in one way or another." (p. 102). Bybee (2005) takes a middle position, citing Erman and Warren (2000) who "found that what they call prefabricated word combinations constitute about 55% of both spoken and written discourse"; Sag et al. (2001) offer comparable estimates.

As regards the number of ELUs used in the language community or belonging to the linguistic competence of native speakers of a language, Jackendoff (1997, 157) quotes Weinreich (1969) citing estimates of over 25,000 fixed expressions stored in the lexicon of an average speaker of English, whereas Mel'čuk reportedly claims the "phrasal lexicon" to be one order of magnitude larger than the word lexicon (Kuiper 2004). Anyway, the conclusion in Jackendoff (1997, 157) that "[t]here are too many idioms and other fixed expressions for us to simply disregard them as phenomena 'on the margin of language'" seems entirely warranted.

2 ELUs on a cline

We will not try to give a definition of ELUs here, other than the one already given above. In the literature one finds discussion of comparable concepts, such as fixed expressions ("vaste verbindingen" (Everaert 1993)) and Multi Word Expressions (Sag et al. 2001, Odijk 2003). A definition such as Everaert's, given below in (1) in the translation of Villada Moirón (2005, p. 2–3), is typical in at least three aspects: 1. it is complex; 2. it contains at least one disjunction; and 3. it refers to undefined or useless concepts such as compositionality (Zadrony 1994):[2]

(1) A combination of two or more words that must at least satisfy the (a) condition and perhaps, but not necessarily, condition (b) and/or (c):
(a) the word combination is fixed;

[1] Other arguments for the importance of ELUs in language use can be found in Wray (2002).

[2] Whether or not our ELUs can be equated to constructions, a classical concept that has gained popularity again in recent sub-branches of linguistics (cf. Fillmore, Kay and O'Connor (1988), Goldberg (1995), Croft (2001), etc.) remains to be seen. A very general definition such as the following probably covers most of our ELUs as well: "It is safe to say [...] that in essence a construction is a pattern in the formal properties of a language (i.e., in its form) that is associated with a particular function. While various theories may choose to interpret this definition broadly or more narrowly, the basic notion of a construction as a pattern of form and function remains the same" (Goldberg and Casenhiser n.d.).

(b) the combination as a whole has a non-compositional or partially compositional meaning;
(c) the syntactic/morphological behavior of the fixed expression and/or its parts is not to be expected given the syntactic/morphologic behavior of the individual words or the combination as a whole

The reason that a proper, exhaustive definition of ELUs is so problematic is that we are actually confronted with a large number of structurally different things. Expressions can be stored in the lexicon because of their noncompositional semantics, their unusual syntax, unexpected pragmatics, unexpected phonology, etc. It is therefore hard, if not impossible, to find a common basis on which ELUs can be categorized (cf. also Wray (2002)).

For the analysis of ELUs, we propose an approach that is based on Jackendoff's constructionist ideas of the organization of the lexicon (Jackendoff 1997). Jackendoff does not restrict the lexicon to word-sized elements, but counts lexically underspecified patterns – constructions – as lexical items as well. In accordance with basic Construction Grammar[3] (CxG) assumptions (cf. also Kay (2002), Goldberg (1995)), we see the lexicon as a hierarchically ordered inheritance network, with a cline of lexical fixedness.

We assume that, at least for computational applications, a categorization of ELUs on the basis of their lexical fixedness may be useful. Every (type of) ELU is given a particular place in the network of constructions that is assumed to be (a model of) our mental lexicon. A prominent advantage of this categorization is that ELUs are integrated into the lexicon in a very straightforward way, namely without giving them a special status compared to simple words or schematic phrases. This seems reasonable if one wants to account for the fact that the larger part of our daily language is made out of fixed expressions, prefabs, etc. Another advantage is the fact that thinking in terms of a cline doesn't lead to categorization problems when it comes to fuzzy boundaries. But let us take a look at the design of such a system first.

One end of our cline of lexically fixed constructions covers those lexical elements that are completely instantiated. This group includes all morphemes and single words that show no morphological variation, such as function words, but also elements that can be referred to as *words with spaces*. Their common characteristics is complete inflexibility, and we therefore have to assume that they are stored in the lexicon as such. Examples for Dutch words-with-spaces are named entities (*United Nations, Kofi Annan*) and prepositional phrases like *op grond van* (on ground of) 'on the basis of' and *ter ere van* (to-the honor of) 'to honor' (the latter being a frozen archaic expression containing a fossilized case of nominal inflection). These simple constructions are listed "as is", and the features that are listed with them include combinatorial, phonological and semantic information.

Walking further on our cline, we find lexical entries that show inflection but do not allow for other types of alternations. At this position, lexical words like nouns and verbs (simplices and compounds) are stored, but also ELUs that don't allow for

[3]Even if we use the "capital c capital g" notation, we refer to a variety of constructionist approaches rather than the one specific by Fillmore and Kay

syntactic alternation, although they may show inflection. These constructions carry additional information about the inflectional paradigm they partake in.

The next landmark on the cline are idiomatic expressions that still do not allow for lexical variation but that are syntactically flexible. An example is the expression *het loodje leggen*: its semantics is non-compositional (the literal meaning is 'to lay the little lead', the actual meaning is 'to come of badly, to die'), but its syntax is more or less like any old direct object construction:[4]

(2) Ponyclub dreigt loodje te leggen.
Pony-club threats lead-DIM to lay.
'Pony club appears to die.'

(3) Internetbedrijven leggen massaal het loodje.
Internet-companies lay massively the lead-DIM.
'Internet companies massively go bankrupt.'

Although idioms such as *het loodje leggen* are supposedly listed with their complete lexical content, they inherit their alternation patterns from more general constructions (e.g. the transitive construction) higher up the cline in order to allow for phenomena like verbal concatenation. The fact that processes such as passivization are not possible in this case is a particular feature of the more specific construction which has to be specified explicitly as well.

The first real lexical variation is found a bit further up the cline. Here we find constructions that are still rather fixed in their overall design, but allow for lexical alternation in particular slots. An example is the *Subj V er geen NP van*-construction (cf. e.g. Hoeksema (2001a)). It is a negative polarity construction used (in informal speech only) to express that one does not understand or believe something. The construction is syntactically flexible up to a certain degree (but it does not allow for passivization) and has three (not completely flexible) slots: The subject is an agent who has to be able to understand, the verb comes from the semantic paradigm of *understand*, *believe* and *being able to* and the slot in the Direct Object-NP is filled with a member of a rather narrow but semantically hard to define paradigm.[5] The following sentences give examples for this particular construction.

(4) hij wist er geen bal van
he knew there no ball of
'he didn't know a thing about it'

(5) ze snapt er geen flikker van
she understands there no faggot of
'she doesn't understand a thing of it'

[4]Examples found via Google, URLs http://home-1.tiscali.nl/~kuifje/editie150101.htm and http://wijkcent.a2000.nl/wijkcent/krant/editie/2001/01033002/txt/050501.htm (01.12.2004).

[5]Sentential negation may also be "raised", as in *ik denk niet dat hij er iets van snapt* (I think not that he there anything of understands) (I don't think he understands a thing of it) or be incorporated in the subject, as in *niemand begreep er ene flikker van* (nobody understood any faggot of it) 'nobody understood a word of it' (Postma 2001); the last example shows another optional particularity discussed by Postma, viz., a negative polarity variant *ene* of the indefinite article whose unmarked form is *een*.

Even if the interesting slots in this construction are flexible throughout, we still do not consider it to be a schematic idiom (i.e., a idiomatic pattern with underspecified slots like the *way*-construction or the resultative construction), as the fillers of the slots are chosen from narrow paradigms and the possible candidates must be learned along with the idiom.[6]

On the other hand, we consider the so-called *way*-construction as a real schematic idiom; it is exemplified in the following sentences:[7]

(6) Braid virus baant zich een weg door email.
Braid virus *banen-3rdSG* itself a way through email.
'Braid virus makes its way through email.'

(7) Twee bussen boren zich een weg naar het hart van Istanbul.
Two busses drill themselves a way through the hart of Istanbul.
'Two buses make their way to the hart of Istanbul.'

(8) De flits baant zich een gloeiend heet pad door de lucht.
The lightning *banen-3rdSG* itself a red hot path through the air.
'The lightning makes his red hot path through the air.'

(9) een prachtige streek waarin zeven riviertjes zich een pad
a wonderful area where-in seven rivers-DIM themselves a path
kronkelen naar de zee
wind to the sea
'a wonderful area where seven rivers wind to the sea'[8]

This kind of construction differs from the last one by an even bigger degree of lexical flexibility. The *X er geen Y van*-construction fills its slots with elements from a narrow paradigm (either to be defined semantically, or stipulated purely lexically), whereas the *way*-construction allows for much more variation, especially as far as the verb slot is concerned.

Even if there is not a single lexically fixed slot in this construction, we would like to consider it an ELU, too, for at least two reasons. Firstly, there is a prototype that shows statistical significance, namely the combination of the verb *banen* and the nominal head *weg*.[9] Secondly, the semantics of the *way*-construction is highly idiosyncratic (see Verhagen (2003)).

What we expect from a categorization like this is a twofold achievement. On the one hand, we assume that a thorough insight into the structural fixedness of ELUs helps retrieving them from corpora, as the corpus search can be refined to a large degree if possible alternations are accounted for (see also Villada Moirón (2005)). On

[6]Moreover, there are strong collocational effects between the verb and the expressions of minimal quantity (Hoeksema 2001b).
[7]For a thorough description of the Dutch *way*-construction, see Verhagen (2003), for a description of the English counterpart, see Goldberg (1995).
[8]Found on http://www.freewebs.com/maisjo/infooverdestreek.htm (01.02.2005).
[9]Verhagen (2003) provides us with corpus evidence revealing that e.g. more than 50 per cent of the instantiations of this construction are built with *banen* (that occurs in this construction only), whereas the rest is spread over about 20 verbs with little significance.

the other hand, we see prominent advantages in implementing hierarchically ordered inheritance networks (cf. Poß (2005)). As we are investigating into the nature of inheritance and which features are inherited by which kind of ELU, we are in need of a categorization of some sort. The advantages of this kind of approach (no problems with fuzzy boundaries, no special theoretical status of ELUs) weigh heavier than the known disadvantages (like e.g. the still missing definition and the fact that we completely neglect semantics).

3 Retrieving ELUs from a corpus

3.1 The goal

We try and operationalize the question of the division of labor between the grammar and the lexicon by investigating, in a corpus of spontaneous spoken language, the amount of constructs that recur significantly more often than chance predicts. In this respect, we restrict ourselves, at least for the time being, to ELUs that can be defined in a statistical way.

3.2 The corpus

For our investigations, we use the Spoken Dutch Corpus (CGN), a collaborative effort of several Dutch and Flemish universities, funded by both the Dutch and the Belgian government, completed in 2003. The corpus contains almost 1000 hours of continuous speech, which amounts to a little less than 10 million words (Oostdijk 2000). The corpus was intended as a major resource both for linguistic research and speech technology. To serve this dual purpose, it contains text fragments recorded in a wide range of communicative settings: spontaneous face-to-face and telephone dialogues, interviews, debates, news broadcasts, etc. Two-thirds of the material is collected in the Netherlands, one third in the Dutch speaking part of Belgium. It is the largest and most diverse database of spoken Dutch collected so far.

3.3 Some problems

Standard computational techniques from collocation research (Manning and Schütze 1999, ch. 5) are useful to find certain types of ELUs. For example, a program such as Wordsmith Tools[10] has no problems in finding classical collocation types such as fixed prepositions with adjectives, such as *trots* ***op*** 'proud of'. The table below show the most frequent two word clusters with *trots* in the Dutch part of the corpus. The collocation we were looking for ranks first.

(10)

Wordsmith clusters with **trots**		
N	Cluster	Freq.
1	*trots op* 'proud of'	75
2	*heel trots* 'very proud'	15
3	*trots en* 'proud and'	7

[10] http://www.lexically.net/wordsmith/index.html

With these methods, one may also find relations between lexical items that are not in the books of reference. For example, employing the same methods, it has been demonstrated (van der Wouden 2002) that the Dutch complex focus particle *niet eens* 'not even' shows strong collocational effects with, among other things,

- other particles (i.e., high frequency (function) words), such as *nog* 'yet' and *meer* 'anymore':

  (11) dat is nog niet eens zo lang geleden
  that is yet not even so long ago
  'that isn't even that long ago'

  (12) de man luistert niet eens meer
  the man listens not even anymore
  'the man doesn't even listen anymore'

- (high frequency content) verbs such as *weten* 'to know'

  (13) ik weet niet eens wie Judith Bosch is
  I know not even who Judith Bosch is
  'I even don't know who Judith Bosch is'

  (14) die weten niet eens waar Nederland ligt
  they know not even where Netherlands lies
  'they don't even know where the Netherlands are'

- modal auxiliaries, especially *kunnen* 'can'[11]

  (15) dus ik kan niet eens werken als ik zou willen
  so I can not even work if I would want
  'so I can't work even if I wanted'

  (16) kunnen nog niet eens hun naam en adres schrijven
  can yet not even their name and address write
  '[they] don't even know [how] to write their names and addresses'

On the other hand, we can already be sure that certain types of ELUs will be missed by our techniques. Consider, for example, the Dutch verb *krijgen* 'to get'. Not unlike its English translation, it can be found in quite a number of idioms, phraseological idioms, light verb constructions etc. The dictionaries and grammars list tens or even hundreds of these ELUs. Some of these ELUs to be found there hardly sound familiar to native speakers of the language, which may mean that these combinations are obsolete or dialectal or something like that. Many others, however, are recognized immediately by native speakers. A case in point is the invective *krijg de klere* 'drop dead', which

[11] This collocational effect appears to be restricted to one reading/usage of *kunnen*, viz., the dynamic (i.e. non-epistemic, non-deontic) one: 'be able to'.

literally means ‘get the cholera’. The ELU also has a variant *je kunt de klere krijgen* ‘you can get the cholera’, which is just as rude as the imperative form. Although both variants are known to all native speakers we questioned, our corpus techniques will not find them, at least not in the corpus we have chosen to use, as *krijg de klere* occurs only once in it, and *je kunt de klere krijgen* not at all.

The sparse data problem, i.e. the fact that our corpus (and probably anyone’s corpus) is too small to offer statistical evidence for all ELUs, is a real one, just like it is a real problem in all other quantitative approaches to language. However, it remains to be seen yet whether the restriction to ELUs defined statistically will seriously flaw the answer to our fundamental question regarding the division of labor between grammar and lexicon, between computation and storage.

Apart from these false negatives, i.e. real ELUs not found by statistics, our quantitative methodology yields false positives as well. Consider *ja* ‘yes’, which is the most frequent word in the Dutch part of the corpus. According to Wordsmith Tools, the following are among the 10 most frequent multi word clusters involving the string *ja*:

(17)

Frequent clusters with *ja*		
cluster	**rank**	**#N**
ja ja	1	181347
ja ja ja	2	84696
ja ja ja ja	3	34430
oh ja	5	18512
ja dat	6	17721
ja maar	7	16343

Most clusters in this little table qualify as ELUs. The first one, e.g., *ja ja* usually functions as a discourse marker (cf. e.g. Schiffrin (1987)) and may express either consent or doubt, depending on the intonation with which it is pronounced (which we assume to be lexicalized with the ELU). Consider the following conversation fragment (edited slightly for expository purposes):

(18) ja die uh die tante Hennie zit een beetje te miepen hè.
yes that uh that aunt Hennie sits a bit to whine PART
‘that aunt Henny is whining a bit, isn’t she?’

(19) ja die wilde zich laten euthanaseren of niet?
yes she wanted self let euthanatize or not
‘yeah, she wanted to have herself euthanatized, didn’t she?’

(20) ja ja die wil euthanasie tegen die tijd.
yes yes that wants euthanasia against that time
‘sure, in due time she wants euthanasia’

The combination *ja dat* ‘yes that’, however, does not seem to be an ELU. The high frequency of its occurrence is due to the interplay of a number of factors concerning the grammar of Dutch and the organization of Dutch conversations: one can use a

discourse marker such as *ja* to express consent and to take the turn at the same time (Mönnink 1988), and it is good practice (Onrust, Verhagen and Doeve 1993) to start one's turn by referring to a topic salient in the conversation by means of a deictic element such as the demonstrative pronoun *dat* 'that'.

(21) ja dit moet nog gedaan worden
yes this must yet done become
'this has yet to be done'

(22) ja dat klopt
yes that knocks
'yes that's correct'

3.4 Some solutions

For the problem of false negatives, there is no principled solution – increasing the size of your corpus may help you to find statistical evidence for certain combinations, but new hapax combinations will turn up; moreover, the number of false positives increases with the size of the corpus. Various solutions have been proposed to overcome the problem of the false positives involving high frequency function words (Manning and Schütze 1999). One approach is to pass the candidate phrases through a part of speech filter which only lets through those patterns that are likely to be interesting combinations (Justeson and Katz 1995), another one excludes certain words (e.g., high frequency function words) from participating in candidate combinations (Smadja and McKeown 1990), a third one is using more sophisticated statistics (e.g. Krenn and Evert (2001)). A common feature of these approaches is that they all work some of the time, but none of them works 100% all of the time, at least not for all types of ELUs (van der Wouden 2001, Villada Moirón 2005).

We will not propose the ultimate solution here, as we didn't find it. Moreover, we assume that there is no such thing as a unique ultimate solution for the problem of identifying all and only ELUs from a corpus, for the simple reason that ELUs are too heterogeneous for that. Ultimately, they are the surface manifestation of a number of complex phenomena, the result of a variety of interactions of atoms, mechanisms, rituals and habits from grammar, lexicon and the extra-linguistic real world.

This conclusion implies that we will have to combine the existing techniques and heuristics that are on the market, and think of developing and validating new ones. Various types come to mind: the corpus is enriched with Part Of Speech information, and part of it is annotated syntactically: these two annotations layers open new horizons to searching for types of ELUs that cannot be found by simple (or not so simple) string matching and statistics on the raw text.

4 The implementation of ELUs

Another interesting challenge lies in the implementation of ELUs. Idiosyncratic combinations are still a hurdle for parsing and generating that is particularly hard to take. Depending on the type of expression, many implementations vary from inelegant to

impossible. E.g., how does a system deal with expressions that do not conform with general grammatical rules (like *by and large*)? And how does the system get hold of the non-compositional semantics of idioms (like *to spill the beans*)? We will give a short overview of an approach of the analysis of ELUs by Sag et al. (2001) that offers solutions for various kinds of ELUs. However, we will raise the question whether a more integrating approach could be adopted if there is a deeper understanding of what the building blocks of constructions are and how they can be used in a computational system.

4.1 ELUs in HPSG

For an analysis in Head-Driven Phrase Structure Grammar (HPSG), a method of dealing with various kinds of ELUs has been offered by Sag et al. (2001). This approach accounts for various types of non-compositional multiword expressions, at least as long as they have at least one stable lexical item. Sag et al. (2001) come to the conclusion that different kinds of multiword expressions should be analyzed in different ways, depending on their nature. Three different categories are established, each category of expressions is dealt with in its own way.

The so-called *fixed expressions* are those that are completely immutable like *by and large*. They are treated like words-with-spaces, therefore, in an implementation this leads to string-type listing. The *semi-fixed expressions* show a still low degree of flexibility, as they only allow for inflection, variety in the choice of determiners, different reflexive pronouns, etc., but not for variation with regard to new lexical items. This group includes decomposable (*spill the beans*) and non-decomposable idioms (*kick the bucket*), compound nominals (*part of speech*), and proper names. As semi-fixed expressions behave like single parts of speech, they are represented in the lexicon as single items, with pointers to the element(s) that can undergo inflection resp. can be replaced. As opposed to the semi-fixed expressions, the group of syntactically flexible expressions allows for a much higher degree of structural variability. Light-verb constructions (*make a mistake*) and verb-particle constructions (*look up*) belong into this category, as well as structurally rather free decomposable idioms (*let the cat out of the bag*). For this heterogeneous group, the different analyzing techniques range from subcategorization (e.g. the verb *hand* subcategorizes for *out*) to a so-called idiomatic-construction analysis.[12]

With these techniques, Sag et al. (2001) can cover a big part of the range of fixed and semi-fixed expressions. What they cannot deal with is the kind of phenomena that we referred to as schematic idioms. As soon as there is no direct lexical trigger in the expression, the method does not work anymore.

A solution within HPSG (Pollard and Sag 1994) could be found in defining a lexical rule that transfers certain intransitive verbs into verbs triggering the *way*-

[12]The term idiomatic construction does not completely cover what it usually refers to in Construction Grammars. The difference (within HPSG) is an analysis where listemes do not get an idiomatic meaning assigned and then, in turn, subcategorize for other listemes with an idiomatic meaning. Instead, the elements are allowed to combine regularly, and the complex expression carries the idiosyncratic meaning. Condition for this is that every listeme in the idiom is known as such.

construction. The problem that we see, at least for the time being, lies in the fact that little is known about the further restrictions of the verbs.[13] Riehemann and Bender (2001) argue in favor of a construction-based rather than a strictly lexicalist approach when it comes to idiosyncratic patterns without a single stable lexical item. We want to go further and investigate into an approach that is entirely based on constructions, no matter if there is too little lexical information or not.

4.2 A Construction Grammar-based approach

Starting point of our approach are the assumptions that underlie the various streams of Construction Grammar (as e.g. Goldberg (1995), Croft (2001), Fillmore et al. (1988)). Langacker (2003) gives an overview of the shared notions within the various traditions. They boil down to non-derivationality, monostratality, unity of grammar and lexicon, a cline from specific to schematic constructions that are all stored in the so-called constructicon (Goldberg (2003)), the linking of constructions in an inheritance network, and unification as the motor that drives composition. The fact that lexical as well as phrasal entries are assumed to be stored in the mental lexicon offers an interesting alternative design of lexicon and grammar rules, compared to the lexicalist HPSG approaches.[14] Thus, the innovative point a system based on constructions may offer is a lexicon where apart from lexically specified entries, underspecified patterns are stored. Moreover, in the Construction Grammar framework, ELUs are treated exactly the same way and receive the same status as words (on the one end of the cline) or abstract schemas (on the other end of the cline). But how can these Construction Grammar tenets help us with the implementation of ELUs? In order to illustrate our approach, we give an analysis of the *way*-construction below.

Verhagen (2003, 34) presents a schematic description of the Dutch *way*-construction:

(23)

$$\left[\begin{array}{lccccc} \text{Sem:} & \text{creator} & \text{create-move,} & \text{for-self} & \text{created-way,} & \text{path} \\ & | & | & | & | & | \\ \text{Syn:} & [\text{SUBJ}_i & [\text{V} & [\text{REFL}_i & [\text{DO}] & \text{OBL}]]] \end{array}\right]$$

This formalization is in accordance with the basic CxG tenet that formal and semantic features of a construction go hand in hand. What the figure shows is the following: the (abstract) construction is built up from meaning components that must occur in all particular constructs. The syntactic structure is obligatory as well: a ditransitive pattern with an oblique argument (i.e., a PP, usually). The two sides of the description have no distinctive power in themselves. The semantic structure could also be

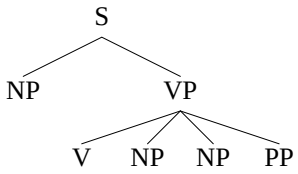
[13] This, of course, is not a proper argument, as every construction has to be analyzed carefully, anyway. Nevertheless, this part of the project aims at designing an experience model rather than a wide coverage grammar of spoken Dutch. The underlying question is not a technical, but rather a cognitive one: Which parts are built from smaller items and which are just bigger building blocks glued together. And for the latter: Which elements can be altered, and which alternations are taken care of by more global mechanisms.

[14] At least the strict lexicalist ones, following the standard approach of Pollard and Sag (1994). In more recent literature, the notion of construction and phrasal patterns rises, see e.g. Sag, Wasow and Bender (2003).

represented by an utterance that does not make use of the way-construction, and a ditransitive sentence with an oblique argument is nothing particularly special, either. What makes it unique is the linking of those two layers (i.e., the lines between the semantic and syntactic structure, showing which semantic component is expressed by which constituent or vice versa).

If we take Verhagen's analysis and translate it into information that can be recognized by a computational system, at least the following reliable information is available: The parser must find a parse of the following structure:

(24)



If the parser recognizes grammatical functions as well, the first NP must be the subject, the second NP must be the indirect object, and the third NP must be the direct object. This is the syntagmatic information. But there is paradigmatic (or even lexical) information available, too. The indirect object NP must be instantiated by a member of the paradigm of reflexive pronouns and cannot be modified, furthermore the pronoun must show agreement with the number and gender features of the subject NP. The direct object NP carries specific lexical information, namely that the determiner must be indefinite. Additional semantic information is available as well, namely that the NP must have PATH-semantics.[15] In the process of choosing the right construction, this information narrows down the list of possible candidates to a rather small set. The PP adjunct must be a directional adverbial, and directionality is caused by the semantics of the preposition. If we store all reliable information in a schema, we get the following picture:

(25)

$$\left[\begin{array}{llllll} \text{Sem:} & & & \text{refl} & \text{way,} & \text{dir} \\ & & & | & | & | \\ \text{Syn:} & [\text{SUBJ}_{\text{i}} & [\text{V} & [\text{IO}_{\text{i}} & [\text{DO}_{-\text{def}}] & \text{PP}]]] \end{array}\right]$$

Note at this point that this new schema is not a replacement for the one given in (23) above, it only mirrors the information that can be retrieved by a parser using conventional methods and that enables the system to distinguish this pattern from all other possible patterns. If the parser finds these features in the input sentence, it may categorize the sentence as an instantiation of the *way*-construction. Once a construction is recognized, all semantic and combinatorial features that its lexicon entry is enriched with must be applied.

[15] In order to be able to process this information, there are two possible ways. Either the lexical items are organized in fixed sets of semantic groups, according to their relevant features, or a semantic ontology must be included into the system. For the time being, we stick to the first solution (with pleasing results), although the latter seems to be attractive as well.

4.3 Putting the puzzle together

The kind of approach we propose differs from lexicalist approaches in the sense that it assumes special phrasal entries rather than special lexical entries. According to constructionist approaches, the (idiosyncratic) semantics of any utterance (and therefore of any ELU)[16] is the contribution of the particular construction, in the first place. Structural information is used for recognition, but additional features are needed to identify a given construct (i.e. the instantiation of a construction).

The features in question range from strictly lexical to abstract semantic. For items like the verb *banen*, for instance, it is economical as well as elegant to adopt a pointer to the *way*-construction in the feature structure of the lexical entry, as *banen* is a hapax in the sense that it cannot occur outside the *way*-construction. But as there are instantiations without the specific lexical items, another way to trigger the construction is proposed in analyzing a bundle of features. In the case of the *way*-construction, the features are basically phrasal and semantic.[17] The in-depth analysis of more constructions will provide us with a deeper insight of the nature of the features that establish constructions, and de design of a system that deals with ELUs on a constructional basis. On a more theoretical level, we expect to find interesting insights regarding the design of the Construction Grammar theory by formalizing it to a degree that makes it implementable.

5 Summary

In this paper, we presented first results of the ongoing research project *Dutch as a Construction Language*. We described how we try to investigate the purely theoretical question of the division of labor between lexicon and grammar using computational methods, namely extraction on the one hand, and implementation on the other. When it comes to extraction, we found that there is not one sing2le statistical method that can cover the whole range of different phenomena we consider an ELU. Different from that, for implementation, we hope to find a useful technique that is inspired by Construction Grammar assumptions and that is able to handle the whole range of constructions using one single mechanism, namely the analysis of feature bundles as phrasal patterns.

References

Altenberg, B.(1998), On the phraseology of spoken English: The evidence of recurrent word-combinations, *in* A. Cowie (ed.), *Phraseology. Theory, Analysis, and Applications*, Clarendon Press, Oxford, pp. 101–22.

Bybee, J.(2005), The impact of use on representation: grammar is usage and usage is grammar, Presedential address LSA, January 8, 2005.

Croft, W.(2001), *Radical Construction Grammar. Syntactic theory in typological perspective*, University Press, Oxford.

[16] Not every construction is an ELU, but probably every ELU is a construction

[17] Semantic in terms of the lexical items found in the construction. Therefore, it does not get in conflict with the categorization of ELUs proposed above, as it still does not make use of the overall semantics of an ELU.

Erman, B. and Warren, B.(2000), The idiom principle and the open choice principle, *Text, an interdisciplinary journal for the study of discourse* **20**(1), 29–62.

Everaert, M.(1993), Vaste verbindingen (in woordenboeken), *Spektator* **22**(1), 3–27.

Fillmore, C. J., Kay, P. and O'Connor, M. C.(1988), Regularity and idiomaticity in grammatical constructions: the case of *let alone*, *Language* **64**(3), 501–39.

Goldberg, A.(2003), Constructions: a new theoretical approach to language, *Trends in Cognitive Sciences* **7**(5), 219–223.

Goldberg, A. E.(1995), *Constructions. A Construction Grammar approach to argument structure*, University Press of Chicago, Chicago.

Goldberg, A. E. and Casenhiser, D.(n.d.), English constructions, Ms. Princeton University, to appear in *Handbook of English Linguistics*. Blackwell Publishers.

Hoeksema, J.(2001a), Partitivity, degrees and polarity, *Verbum* **XXV**(1), 81–96.

Hoeksema, J.(2001b), Rapid change among expletive polarity items, *in* L. J. Brinton (ed.), *Historical Linguistics 1999. Selected Papers from the 14th International Conference on Historical linguistics, Vancouver, 9–13 August 1999*, John Benjamins, Amsterdam/Philadelphia, pp. 175–186.

Jackendoff, R.(1997), *The Architecture of the Language Faculty*, The MIT Press, Cambridge, Mass.

Justeson, J. S. and Katz, S. M.(1995), Technical terminology: some linguistic properties and an algorithm for indentification in text, *Natural Language Engineering* **1**, 9–27.

Kay, P.(2002), An informal sketch of a formal architecture for construction grammar, *Grammar* **5**, 1–19.

Krenn, B. and Evert, S.(2001), Can we do better than frequency? A case study on extracting PP-verb collocations, *in* B. Daille and G. Williams (eds), *COLLOCATION: Computational Extraction, Analysis and Exploitation. Proceedings of a Workshop during the 39th Annual Meeting of the Association for Computational Linguistics and the 10th Conference of the European Chapter, Toulouse, France, July 7th*, CNRS – Institut de Recherche en Informatique de Toulouse, and Université de Sciences Sociales, Toulouse, France, pp. 39–46.

Kuiper, K.(1996), *Smooth talkers: the linguistic performance of auctioneers and sportscasters*, Lawrence Erlbaum Associates, Mahwah, NJ.

Kuiper, K.(2004), [review of] A. Wray: Formulaic language and the lexicon, *Language* **80**(4), 868–872.

Langacker, R. W.(2003), Construction grammars. cognitive, radical and less so, Plenary Paper ICLC 8, Logroño, Spain.

Manning, C. D. and Schütze, H.(1999), *Foundations of statistical natural language processing*, The MIT Press, Cambridge, Mass.

Mönnink, J.(1988), *De organisatie van gesprekken: een pragmatische studie van minimale interaktieve taalvormen*, PhD thesis, Nijmegen.

Odijk, J.(2003), Towards a standard for multi-word expressions. ISLE Project Report, February 2003, http://lingue.ilc.cnr.it/EAGLES96/isle/clwg_doc/ISLE_D6.1.zip.

Onrust, M., Verhagen, A. and Doeve, R.(1993), *Formuleren*, Bohn Stafleu Van Loghum, Houten.

Oostdijk, N.(2000), The Spoken Dutch Corpus. Overview and first evaluation, *in* M. Gavralidou, G. Carayannis, S. Markantonatou, S. Piperidis and G. Stainhaouer (eds), *Proceedings of the second International Conference on Language Resources and Evaluation*, ELRA, Paris, pp. 887–893.

Pollard, C. and Sag, I. A.(1994), *Head-driven Phrase Structure Grammar*, University Press of Chicago, Chicago.

Poß, M.(2005), Towards an implementation of constructions, *Proceedings of the Sixth Annual High Desert Linguistics Society Conference*. to appear.

Postma, G.(2001), Negative polarity and the syntax of taboo, *in* J. Hoeksema, H. Rullmann, V. Sánchez Valencia and T. van der Wouden (eds), *Perspectives on Negation*, John Benjamins, Amsterdam, pp. 283–330.

Riehemann, S. Z. and Bender, E.(2001), Absolute constructions: On the distribution of predicative idioms, *in* S. Bird, A. Carnie, J. D. Haugen and P. Norquest (eds), *WCCFL 18 Proceedings*, Cascadilla Press, Somerville, pp. 476–89.

Sag, I. A., Baldwin, T., Bond, F., Copestake, A. and Flickinger, D.(2001), Multiword expressions: A pain in the neck for NLP, LinGO Working Paper No. 2001-03 (CSLI Linguistic Grammars Online (LinGO) Lab, Stanford University); also in Proceedings of the Third International Conference on Intelligent Text Processing and Computational Linguistics (CICLING 2002), Mexico City, Mexico, pp. 1-15.

Sag, I. A., Wasow, T. and Bender, E.(2003), *Syntactic Theory: A formal introduction*, 2 edn, CSLI, Stanford.

Schiffrin, D.(1987), *Discourse markers*, Cambridge University Press, Cambridge.

Smadja, F. A. and McKeown, K. R.(1990), Automatically extracting and representing collocations for language generation, *Proceedings of the 28th Annual Meeting of the Association for Computational Linguistics*, pp. 252–259.

Sprenger, S. A.(2003), *Fixed expressions and the production of idioms*, PhD thesis, Nijmegen.

van der Wouden, T.(2001), Collocational behaviour in non content words, *in* B. Daille and G. Williams (eds), *COLLOCATION: Computational Extraction, Analysis and Exploitation. Proceedings of a Workshop during the 39th Annual Meeting of the Association for Computational Linguistics and the 10th Conference of the European Chapter, Toulouse, France, July 7th*, CNRS – Institut de Recherche en Informatique de Toulouse, and Université de Sciences Sociales, Toulouse, France, pp. 16–23.

van der Wouden, T.(2002), Particle research meets corpus linguistics: on the collocational behavior of particles, *in* T. van der Wouden, A. Foolen and P. Van de Craen (eds), *Particles. Belgian Journal of Linguistics 16*, John Benjamins, Amsterdam, pp. 151–174.

Verhagen, A.(2003), The Dutch *way*, *in* A. Verhagen and J. van de Weijer (eds), *Usage-Based Approaches to Dutch. Lexicon, grammar and discourse*, Lot, Utrecht, pp. 27–57.

Villada Moirón, B.(2005), *Data-driven Identification of Fixed Expressions and their Modifiability*, PhD thesis, Groningen.

Weinreich, U.(1969), Problems in the analysis of ididioms, *in* J. Puhvel (ed.), *Sub-*

stance and structure of language, University of California Press, Berkeley and Los Angeles, pp. 23–81. (reprinted in *On Semantics*, 1980).

Wray, A.(2002), *Formulaic Language and the Lexicon*, Cambridge University Press, Cambridge.

Zadrony, W.(1994), From compositional to systematic semantics, *Linguistics and Philosophy* **17**, 329–342.

acknowledgement

The research reported on here is carried out within the framework of the VIDI-project *Dutch as a construction language*, financed by NWO, the Dutch Organization for Scientific Research (project number 276-70-003), and Leiden University.

Preference-Driven Bimachine Compilation

An Application to TTS Text Normalisation

Wojciech Skut

Rhetorical Systems

Abstract

This paper describes a grammar formalism and a deterministic parser developed for text normalisation in the rVoice[1] text-to-speech (TTS) system. The rules are formulated using regular expressions and converted into a non-deterministic finite-state transducer (FST). At runtime, search is guided by parsing preferences which the user may associate with regular operators; the best solution is determined in a way similar to the directional evaluation of constraints in Optimality Theory. During compilation, the FST is converted into a bimachine, making deterministic parsing possible.

1 Motivation

Over the past decade, speech synthesis has become one of the most succesful commercial applications of natural language and speech processing technologies. During this time, it has established itself as a standard solution in call centre applications, telephone banking and several other areas. The reason for its success is rightly attributed to the emergence of high-quality unit selection synthesis methods (Hunt and Black 1996). However, as a result of this focus on *speech* processing, the *text* processing part of TTS typically receives less attention although a high quality text front end is indispensable for a good TTS system.

The task of the text front end in a TTS system is to convert raw input text into a sequence of unambiguous phonetic symbols. For example, the sentence *"On 22/5, Mr Brown had to pay an $80 fine"* may be transformed to the phonetic representation [on ðə twentɪ sekənd əv meɪ mɪstə braʊn hæd tə peɪ ən eɪtɪ dolə faɪn]. While mapping some of the words (e.g. *pay*) to phonetic symbols is straightforward, other parts of the input require a complex multi-stage transformation, e.g., *$80 → eighty dollar → [eɪtɪ dolə].* Non-trivial processing is required for numbers, dates (*22/5*), currency amounts (*$80*), abbreviations (*Mr*) and many other types of expressions. It is often context dependent (e.g. *$80 → eighty dollar/eighty dollars* as in *an $80 fine* vs. *he was fined $80*).

Typically, text processing is split into two main stages. In the first one, abbreviations, digits and symbols are rewritten as literal text (e.g. *On 22/5, Mr Brown had to pay an $80 fine → On the twenty-second of May, Mister Brown had to pay an eighty dollar fine*). Then, the actual phonetic rewrite takes place.

The term *text normalisation* commonly refers to the first processing stage, which is also the more difficult one. Due to the required broad coverage, text normalisation grammars tend to be very large and complex. As in other areas of NLP, disambiguation of alternative analyses is the main source of complexity: the string *1* may be expanded

[1] See `http://www.rhetorical.com/cgi-bin/demo.cgi`.

as *one*, *first*, *premier*, etc. depending on the context it appears in. Although it may be possible to learn such context-based disambiguation rules from data, existing text normalisation systems are typically rule-based, which is most probably due to two reasons.

Firstly, existing rule-based formalisms often come with very large and detailed grammars developed over years, an extremely valuable resource that might be wasted if one decided to abandon the rule-based paradigm. Secondly, text normalisation requires very high precision as we cannot afford, say, an account balance to be expanded incorrectly in an automated telephone banking application. The precision threshold is thus much higher than in those areas of NLP where data-driven approaches have become successful, e.g. in Machine Translation. This does not preclude a data-drived solution to the problem, but the quality restrictions imposed on the potential solutions make it very hard.

In effect, this paper focuses on disambiguation strategies for purely rule-based grammars. It shows how to implement a simple but intuitive and powerful disambiguation strategy within a system that is both efficient at runtime and expressive enough to handle typical text normalisation constructs. The use of the *finite-state transducer* (FST) framework provides for a good compromise between expressive power and computational tractability: a grammar is first compiled into a non-determinstic FST containing *markers* that express user-defined local parsing preferences. In an extra step, the preferences are used to turn the FST into a deterministic device called *bimachine*. The system scales well to large grammars and supports efficient and ergonomical grammar development, including interactive rule compilation and debugging.

2 Definitions and Notation

The following definitions are intended to clarify the notation for some basic finite-state constructs. The less well known notion of *bimachines* is introduced and explained in section 5.

Definition 1. *(***Nondeterministic FSA***) A* non-deterministic finite-state automaton *(NFSA) over input alphabet Σ is a quintuple $A = (\Sigma, Q, q_0, E, F)$, where Q is the set of states of A, $q_0 \in Q$ is the initial state, and $F \subset Q$ the set of final states, and $E \subset Q \times (\Sigma \cup \{\epsilon\}) \times Q$ the set of A's transitions.*

Definition 2. *(***Deterministic FSA***) A* deterministic finite-state automaton *(DFSA) is a quintuple $A = (\Sigma, Q, q_0, \delta, F)$, where $q_0 \in Q$ is the initial state, $\delta : Q \times \Sigma \rightarrow Q$ the transition function, and F the set of final states. The symbol δ^* is used to denote the extension of the transition function to the domain $Q \times \Sigma^*$: $\delta^*(q, \epsilon) := q$, $\delta^*(q, ua) := \delta(\delta^*(q, u), a)$ for $a \in \Sigma, u \in \Sigma^*$.*

Definition 3. *(***Finite-State Transducer***) A* finite state transducer *(FST) $T = (\Sigma, \Delta^*, Q, q_0, E, F)$ over an input alphabet Σ and output alphabet Δ is defined identically to an NFSA except that each transition contains an output label, i.e., $E \subset Q \times (\Sigma \cup \{\epsilon\}) \times Q \times \Delta^*$. We will use the notation $e.source$, $e.input$, $e.output$ and $e.target$ to refer to the respective components of a transition $e \in E$.*

Definition 4. *(***Sequential Transducer***) A* sequential transducer *is the two-level counterpart of a DFSA. It is a 7-tuple* $T = (\Sigma, \Delta, Q, q_0, \delta, \sigma, F)$*, such that* $(\Sigma, Q, q_0, \delta, F)$ *is a DFSA, and* $\sigma : Q \times \Sigma \rightarrow \Delta^*$ *is the* output function *defining the output of each transition.*

Definition 5. *(***Path, Reachability***) A path in an NFSA/FST is a sequence of transitions* $\pi = e_1, \ldots, e_t$ *such that* $e_j.target = e_{j+1}.source$ *for* $j = 1, \ldots, t-1$*. A path* π *consumes a string* w *if* $e_1.input \cdot \ldots \cdot e_t.input = w$*. In the following, the term path always denotes an* ϵ-cycle-free path*. In analogy to transitions, we will also write* $\pi.source$*,* $\pi.target$*, etc.*

The notation $q \stackrel{a:o}{\Rightarrow} q'$ *indicates that state* q' *is reachable from state* q *by exactly one transition consuming symbol* $a \in \Sigma$ *and emitting* o*, i.e.,* $(q, a, q', o) \in E$*. For a string* $u \in \Sigma^*$*,* $q \stackrel{u:o^*}{\Rightarrow} q'$ *denotes the reflexive-transitive closure of* $\Rightarrow$*, i.e., there is a path in* T *from* q *to* q' *consuming input* u *and emitting* $o \in \Delta^*$*.*

3 Grammars

Text normalisation consists of two sub-tasks: *parsing* (the identification of constructions that require normalisation) and *rewriting* (the actual string transformations, e.g. *$11 → eleven dollars*). Therefore, the formalism is split into *parsing rules* and *rewrite rules*.

The syntax of parsing rules is roughly based on common parser generators such as `yacc` or `bison`.[2] Each rule consists of a left-hand side non-terminal symbol (the *rule name*), a right-hand side specifying its *expansion* as a regular expression over terminal and non-terminal symbols, and an optional *rewrite statement* that calls *rewrite rules* in order to perform string rewriting, as shown below:

```
date
    ->
      day:$D [name="/"] month:$M
                      ([name="/"] year:$Y)?
 {"the" exp_ordinal($D) "of" $M exp_year($Y)};

day -> [name="(0?)[1-9]|[12][0-9]|3[01]"];

month -> [name="(0?)[1-9]|1[012]"];

year -> [name="[12][0-9][0-9][0-9]"];
```

Here, the left-hand side symbol `date` is expanded to `day` followed by the terminal `/`, the nonterminal `month`, and an optional instance of the nonterminal `year` preceded by the separator `/`. The symbols `day`, `month` and `year` expand to terminals according to the respective rules. Each terminal is a simple feature structure that denotes a token (in the above grammar fragment, only the feature `name` is used).

[2]Note, however, that the semantics of the rules, as discussed below in section 4.2, is different: while LR grammars such as the ones accepted by `yacc` and `bison` are expected to be unambiguous, our formalism allows a substantial amount of ambiguity in the grammar.

The `date` rule is also associated with a rewrite statement enclosed in curly brackets. It specifies that a date, e.g. `13/05/2001`, should be expanded into the string *the thirteenth of May two thousand and one*. The symbols `$D`, `$M` and `$Y` are coreferences between constituents of the right-hand side of a rule and the associated rewrite statement. The functions `exp_ordinal()` and `exp_year()` are *rewrite rules* that specify how a constituent should be rewritten in the normalisation process. The grammar may also insert new tokens (as it does `"the"` and `"of"`) as well as reorder or duplicate existing ones.

Parsing rules can optionally be associated with left and right context restrictions. For example, the rule `A → B / C D / E` will expand the sequence `C D` to `A` only if it is preceded by a match of regular expression `B` and followed by a match of regular expression `E`.

The *rewrite rules*, formulated in a two-level regular calculus, do not introduce any novel constructs, so they are not discussed in detail.

The expressive power of the formalism is restricted to a regular language by a) excluding recursion of the form $X \rightarrow \ldots \rightarrow X$[3] and b) an implicit treatment of constituent reordering and duplication, which are non-regular operations. The grammar is converted to an FST translating input strings to possible constituent bracketings including special markers for the non-regular operators. At runtime, the selected bracketing is converted to a tree, on which the extra operations are performed. The constituents of the reordered tree are then sent to the rewrite grammar according to the rewrite rules specified in the rewrite statements of the respective parsing rules.

The grammar FST is typically ambiguous, i.e. it encodes a non-functional rational relation, and may return more than one analysis for an input string. Thus, the actual challenge is to devise a disambiguation strategy with a simple and user-friendly semantics that would make it easier for the grammar developer to maintain control over the behaviour of the grammar.

4 Finite-State Parsing Preferences

As a first step, we shall see how to incorporate parsing preferences into a finite-state grammar. It turns out that preferences can be a) associated with regular operators, and b) translated into an FST in a meaningful way. The result is a prioritisation of alternative paths for a word w in the FST that also corresponds to the order of results in a naïve depth-first search with backtracking. This result will be used in section 5 to establish a deterministic and fail-safe best-first search technique.

4.1 Regular Operators

In a finite-state grammar, complex structures are created from simpler regular expressions using *regular operations*. Their basic inventory comprises *concatenation* (AB, $A \cdot B$), *union* ($A|B$) and *closure* (A^*). Further commonly used operators, such as $?$

[3] This restriction might appear harsh, but text normalisation tasks typically do not require more expressive power (Sproat 1996).

and $+$, can be derived from the primitive ones listed above and need not be considered at first.

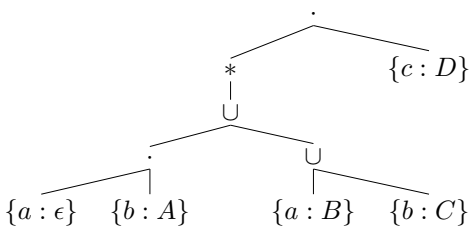


Figure 1: Regular expression $(\{ab : A\}|\{a : B\}|\{b : C\})^*\{c : D\}$ shown as a tree.

The resulting structure forms a tree as shown in figure 1. Such a tree can be compiled into a (non-deterministic) FST. The simplest compilation algorithm (*Thompson's algorithm*) is to create a network of transitions that directly correspond to a traversal of the regexp tree (Hopcroft, Motwani and Ullman 2001). Each node/leaf in the tree translates into two FST states: an *in* state ("we enter the subexpression rooted in the current node") and an *out* state ("we leave the subexpression rooted in the current node"). The *in* state and the *out* state of the root node are, respectively, the initial and the (only) final state of the FST. The states are connected by transitions in accordance with the semantics of the regular operators at the respective tree nodes, as shown in table 1.

Node type	Transitions added
$X = Y\|Z$	$\{X_{in} \overset{\epsilon:\epsilon}{\Rightarrow} Y_{in}, X_{in} \overset{\epsilon:\epsilon}{\Rightarrow} Z_{in}$ $Y_{out} \overset{\epsilon:\epsilon}{\Rightarrow} X_{out}, Z_{out} \overset{\epsilon:\epsilon}{\Rightarrow} X_{out}\}$
$X = Y \cdot Z$	$\{X_{in} \overset{\epsilon:\epsilon}{\Rightarrow} Y_{in}, Y_{out} \overset{\epsilon:\epsilon}{\Rightarrow} Z_{in}$ $Z_{out} \overset{\epsilon:\epsilon}{\Rightarrow} X_{out}\}$
$X = Y^*$	$\{X_{in} \overset{\epsilon:\epsilon}{\Rightarrow} X_{out}, Y_{out} \overset{\epsilon:\epsilon}{\Rightarrow} X_{out},$ $X_{out} \overset{\epsilon:\epsilon}{\Rightarrow} Y_{in}\}$
$X = \{a : o\}$	$\{X_{in} \overset{a:o}{\Rightarrow} X_{out}\}$

Table 1: The FST compilation of union, concatenation, closure and an atomic regexp.

4.2 Local Parsing Preferences

Now suppose we want to find a translation for a string w licensed by a compiled regular expression. A possible, although naïve, strategy is to explore all paths starting from the initial state until we reach the final state. Whenever we find that we have run into a dead end, we backtrack to the previous choice point and explore another

path from there. As long as we omit ϵ-cycles, the algorithm will eventually terminate. Also, if there is more than one translation, the local search decisions taken at the choice points will influence the result.

A simple but powerful disambiguation strategy can be pursued here:

- longest or shortest match (distinguished by notation) at the closure operator;
- exploring disjunction branches in order of their disjuncts.

These two simple rules give the grammar developer full control over all ambiguity in the system. They are also very intuitive: if longest-match is chosen as the default interpretation for the closure operator, the strategy resembles the evaluation of Perl regular expressions, which most users can be assumed to be familiar with.

The reader may object that this disambiguation strategy is *too* simplistic, especially compared to frameworks such as Optimality Theory (OT, (Ellison 1994, Karttunen 1996, Eisner 2000)), which splits linguistic knowledge into a device generating all possible analyses (Gen) and a number of constraints that rank these analyses according to the number and severity of constraint violations. The analysis with the fewest violations wins.

However, this elegant framework is intended as a tool for linguistically adequate theoretical modelling of clear-cut phenomena. It is doubtful that a "dirty" task like text normalisation could be decomposed into a neat hierarchy of constraints filtering the possible analyses. In addition, TTS grammar developers are more likely to be familiar with Perl regular expressions than Optimality Theory. As a matter of fact, the author is not aware of a single large-scale NLP system based on OT.

Nevertheless, we will see that some formal concepts developed in the framework of OT turn out to be very useful for the purpose of preference-based compilation, cf. section 4.4.

4.3 Preference-Driven Search

The simple disambiguation strategy outlined in the previous section can be easily translated to an FST framework by adding some control information to the choice points. Note that only two types of regexp tree nodes introduce non-determinism: union and closure. Thus, if X is a disjunction node and Y, Z are the disjuncts, then the compilation algorithm will create the ϵ-transitions shown in figure 2.

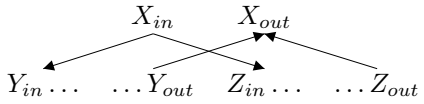


Figure 2: Compilation of the union operator.

The choice point occurs at X_{in}: we can either go to Y_{in} or to Z_{in}. All we need to implement the chosen strategy is to make sure that the move $X_{in} \stackrel{\epsilon}{\Rightarrow} Y_{in}$ will be performed first (i.e. before $X_{in} \stackrel{\epsilon}{\Rightarrow} Z_{in}$).

Figure 3 shows the other case, namely the closure operation. Here, X is the node corresponding to the closure operator, Y the root of the embedded expression and V the parent node of X. The choice point occurs at state X_{out}, from where we can go either to Y_{in} or to V. Note that the former option corresponds to the longest-matchand the latter to the shortest-match strategy.

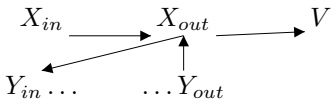


Figure 3: Compilation of the closure operator.

4.4 Encoding

In order to eliminate the non-determinism of the search method sketched above, it is worthwhile to take a closer look at finite-state approaches to Optimality Theory (OT), where a non-deterministic FST (Gen) encodes all possible pairs of inputs (called underlying representations, UR) and outputs (called surface representations, SR). The generated SRs are evaluated by applying a sequence of constraint FSTs $(C_1, \ldots, C_n)$, which either filter out some of the possible SRs or insert *markers* denoting the fulfillment/violation of a constraint. The markers can be used to guide the search for the optimal solution.

Out of all types of OT constraints, the strategy described in the previous section resembles *left-to-right directional constraints* (Eisner 2000). Following Eisner, whenever two possible transitions labelled with the same input symbol $a \in \Sigma \cup \{\epsilon\}$ leave a state q, the preferred one is assigned the mark m_0 and the less preferred one the mark m_1. In the case of the compilation method specified in section 4.1, such choice points are always binary and are possible only for $a = \epsilon$. Thus, the simplest encoding is to set $e.output = m_0$ or respectively $e.output = m_1$ for all such ϵ-transitions leaving choice point states.

In this way, each path $\pi = e_1, \ldots, e_n$ in a transducer T can be associated with a sequence of preference marks $\omega = \omega_1 \ldots \omega_k$, which can be extracted from $\pi.output$ simply by ignoring all output symbols other than m_0 and m_1. Let such a sequence be denoted $\pi.score$.

Note that the relative preference order of two paths $\pi^{(1)}$ and $\pi^{(2)}$ is expressed by the lexicographic order of the mark sequences:

$$\pi^{(1)} \prec \pi^{(2)} \iff \pi^{(1)}.score <_{lex} \pi^{(2)}.score$$

Generalised to sets Π of paths accepting a string w, this criterion states that the preferred path is the first one in lexicographic order:

$$\pi_{best} = min_{\prec}(\Pi)$$

In particular, if $\Pi_T(w)$ denotes the set of all paths in T that consume $w \in \Sigma^*$, then the preferred translation for w is the output of the path $\pi_{best}(w) \in \Pi_T(w)$ defined as:

$$\begin{aligned}\pi_{best}(w) \quad = \quad & min_{\prec}(\{\pi \in \Pi_T(w) : \\ & \pi.source = q_0 \wedge \pi.target \in F\})\end{aligned}$$

Note that we may want to make transducer T ϵ-free for further processing stages. This is not a problem because the preference order of paths is preserved in ϵ-elimination.

5 Search Strategy

Let $\hat{T} = (\Sigma, \Delta, \hat{Q}, q_0, \hat{E}, \hat{F})$ be an ϵ-free FST constructed from a regular expression using the compilation method described in section 4.1 and some ϵ-elimination algorithm. Given an input string $w = a_1 \dots a_t$, we want to find the best-scoring successful path for w in $\hat{T}$.

Note that if the unsuccessful paths are pruned away in advance, the search boils down to a chain of purely local decisions. We start by choosing the lowest-score arc (q_0, a_1, q_1, o_1) starting in q_0, and then choose the lowest-score transition starting in q_1 and accepting a_2, etc.

The set of all successful paths for w can be determined in time $O(t)$ using the following factorisation method. Let $\hat{E}_{DFSA} = \{\langle q, a, q' \rangle : \exists o \in \Delta^* : \langle q, a, q', o \rangle \in \hat{E}\}$ be the projection of the transitions of $\hat{T}$ onto the first three components (i.e. source state, input symbol and output symbol). We determinize the NFSA $A = (\Sigma, \hat{Q}, q_0, \hat{E}_{DFSA}, \hat{F})$ for the input language of $\hat{T}$ using a variant of the subset construction algorithm (Hopcroft et al. 2001). This operation yields a) a DFSA $\overrightarrow{A} = (\Sigma, \overrightarrow{Q}, \overrightarrow{q}_0, \overrightarrow{\delta}, \overrightarrow{F})$ accepting $\mathcal{L}(\hat{T})$ and b) a function $\overrightarrow{h} : \overrightarrow{Q} \rightarrow 2^Q$ mapping each state of the DFSA to the corresponding set of states of $\hat{T}$. More precisely, $\overrightarrow{h}(\overrightarrow{\delta}(\overrightarrow{q}_0, u))$ is the set of states reachable in $\hat{T}$ from q_0 by consuming the string u.

In a similar way, we can construct an acceptor $\overleftarrow{A} = (\Sigma, \overleftarrow{Q}, \overleftarrow{q}_0, \overleftarrow{\delta}, \overleftarrow{F})$ and a function $\overleftarrow{h}$ by reversing and determinising A. Accordingly, $\overleftarrow{h}(\overleftarrow{\delta}(\overleftarrow{q}_0, v^{-1}))$ is the set of all states $q \in \hat{Q}$ such that $q \stackrel{v}{\Rightarrow} F$.

If now $w = uv \in \Sigma^*$, then the intersection $\overrightarrow{h}(\overrightarrow{\delta}(\overrightarrow{q}_0, u)) \cap \overleftarrow{h}(\overleftarrow{\delta}(\overleftarrow{q}_0, v^{-1}))$ is the set of all states on a successful path in $\hat{T}$ accepting v after consuming the prefix u. Thus, given a string $w = a_1 \dots a_t$, we can determine the successful paths in $\hat{T}$ in time $O(t)$ by:

- running $\overrightarrow{A}$ on string w and keeping the path $\overrightarrow{q}_0 \overrightarrow{q}_1, \dots, \overrightarrow{q}_t$;
- running $\overleftarrow{A}$ on w^{-1} and keeping the path $\overleftarrow{q}_0 \overleftarrow{q}_1, \dots, \overleftarrow{q}_t$;
- forming a sequence of *reachability sets* $R_0, \dots, R_t$ constructed as follows: $R_j = \overrightarrow{h}(\overrightarrow{q}_j) \cap \overleftarrow{h}(\overleftarrow{q}_{t-j})$, $j = 0, \dots, t$.

Since T is non-deterministic, there are several alternative paths $\pi^{(i)} = e_1^{(i)}, \dots, e_t^{(i)}$ for w, each associated with a unique score $\pi^{(i)}.score$. The preferred

translation is the output of the first path $\pi^{(i)}$ according to the lexicographic order of the path scores. Thus, the preferred path can be found deterministically in time $O(t)$ according to the following formula:[4]

$$e_j = \begin{cases} min_{\prec}(Out(q_0, a_1)) & j = 1 \\ min_{\prec}(Out(e_{j-1}.source, a_j)) & j > 1 \end{cases} \quad (1)$$

In this way, we arrive at a deterministic parsing strategy that guarantees finding the best parse (according to the locally expressed disambiguation preferences) in time $O(t)$.

The construction of $\overleftarrow{A}$ and $\overrightarrow{A}$ together with a recursive formula for the best path is closely related to *bimachines* (Berstel 1979, Roche and Schabes 1996). A *bimachine* is a triple $B = (\overrightarrow{A}, \overleftarrow{A}, \gamma)$, where $\overrightarrow{A} = (\Sigma, \overrightarrow{Q}, \overrightarrow{q}_0, \overrightarrow{\delta}, \overrightarrow{Q})$ is a left-to-right DFSA, $\overleftarrow{A} = (\Sigma, \overleftarrow{Q}, \overleftarrow{q}_0, \overleftarrow{\delta}, \overleftarrow{Q})$ is a right-to-left DFSA, and $\gamma : \overrightarrow{Q} \times \overleftarrow{Q} \to \Delta^*$ the output function of B. Applied to a string $w = a_1 \ldots w_t$, B outputs the string $b_1 \ldots b_t$, where $b_i = \gamma(\overrightarrow{\delta}^*(\overrightarrow{q}_0, b_1 \ldots b_{i-1}), \overleftarrow{\delta}^*(\overleftarrow{q}_0, b_t \ldots b_i))$.

The importance of bimachines lies in the fact that a) they are deterministic and b) every unambiguous FST — even a non-determinisable one — can be converted to a bimachine. Since most interesting cases of FSTs actually *are* ambiguous, the construction presented in this section extends the notion of bimachines to FSTs which are disambiguated at runtime via preference marks.[5] Accordingly, the output function is not specified explicitly; instead, it follows from the recursive formula (1).

6 Optimisation and Evaluation

6.1 Runtime Optimisation

The algorithm can be made more efficient by means of precompilation. Instead of running $\overrightarrow{A}$ and $\overleftarrow{A}$ separately, and then performing the disambiguation step described by (1), the task can be performed in only two stages.

In the first pass, the acceptor $\overleftarrow{A}$ is run on the reversed input string, producing a path $\overleftarrow{q}_0, \ldots, \overleftarrow{q}_t$. This path is then combined with the original input w in order to form a sequence of state-input pairs of the following form: $\langle a_1, \overleftarrow{q}_{t-1}\rangle, \ldots, \langle a_j, \overleftarrow{q}_{t-j}\rangle, \ldots, \langle a_t, \overleftarrow{q}_0\rangle$. This sequence serves as input to the second component of the system, which is a sequential transducer $\tilde{T} = (\Sigma \times \overleftarrow{Q}, \Delta, \hat{Q}, q_0, \delta, \sigma, \hat{F})$ over the complex input alphabet $\Sigma \times \overleftarrow{Q}$. If w is accepted, $\tilde{T}$ outputs the preferred translation of w. The functions δ and σ are defined as follows.

$$\begin{aligned} \delta(q, \langle a, q'\rangle) &= min_{\prec_{(q,a)}}(\{r \in \overleftarrow{h}(q') : q \stackrel{a}{\Rightarrow}_{\hat{T}} r\}) \\ \sigma(q, \langle a, q'\rangle) &= o(q, a, \delta(q, (a, q'))). \end{aligned}$$

The application of the above device to a string w is deterministic and very efficient: it requires $2|w|$ transition lookup steps; the execution time of each of these steps can be made constant by employing appropriate data structures.

[4] $Out(q, a)$ denotes the set of all transitions leaving q via symbol a. $min_{\prec}$ is well-defined because the transitions $e \in Out(e_{j-1}.source, a_t)$ have distinct scores, i.e., they are totally ordered by $\prec$.

[5] For another application of the concept of bimachine factorisation to ambiguous FSTs, see Kempe (2001).

6.2 Compile-Time Optimisation

There are several possible optimisations aiming at speeding up the compilation process. First of all, the structure $(\overleftarrow{A}, \tilde{T})$ introduced above for runtime optimisation is also faster to compile since its compilation involves only one potentially expensive determinisation step (for $\overleftarrow{A}$) instead of two (for $\overleftarrow{A}$ and $\overrightarrow{A}$).

Another improvement is related to the representation of the markers, which need not be present at runtime. Instead, alternative transitions can be stored in a list in an order corresponding to the relation $\prec$.

6.3 Compilation Speed

As mentioned above, the determinisation of $\overleftarrow{A}$ is the only expensive step in the construction. It took 28 minutes for a grammar fragment comprising 78 rules, while the running time of the remaining compilation steps was under 2 seconds. For larger grammar fragments, the discrepancy was even bigger, indicating scalability problems. Therefore, the decision was taken to introduce two compilation modes: *development* and *release*.

In the development mode, $\overleftarrow{A}$ is not constructed, and the search for the optimal parse involves backtracking. On average, this is around 11 times slower than deterministic search, but the difference is not noticeable to the grammar developer. The benefit is that a grammar can be developed, compiled and tested interactively.

In the release mode, the construction of $\overleftarrow{A}$ is sped up by removing some irrelevant bracketing information from the FST. For the above test fragment, the size of the grammar transducer was thus reduced from 110,056 to 29,231 transitions, and $\overleftarrow{A}$ took only 8 minutes to construct. For the largest grammar available, comprising 214 rules, compilation took 34 minutes, resulting in an FST containing 104,551 transitions (after reduction). $\overleftarrow{A}$ had 671,331 transitions.

The above behaviour of the compiler means it is scalable to medium-sized systems comprising hundreds of rules. For even larger grammars, we contemplate the construction of several preference-based bimachines: one bimachine recognising the matches of the top-level grammar rules (e.g. `date`, `time` or `phone_number`), and one for each of the respective subgrammars.

6.4 Expressive Power

The formalism presented in this paper is a compromise between expressive power and processing efficiency: linear-time parsing is achieved at the expense of keeping the formalism relatively simple. The question is how much the restrictions imposed on the grammar affect the convenience of grammar development.

As far as text normalisation in TTS systems is concerned, the compromise pays off to a large extent. The predecessor of our system, which employed exactly the same disambiguation strategy (longest match plus a left-to-right preference order on disjunctive rules), proved to be expressive enough as a formalism for industrial-scale

multilingual development.[6] The grammar developers have largely found the parsing strategy intuitive and easy to follow — despite some criticism expressed with regard to the global ordering of rules, which was deemed too rigid in certain cases. Hence, the development of larger grammar fragments (partly using automatic conversion from the old grammar formalism), can be expected to be straightforward.

7 Extensions

In section 4.1, the inventory of regular operators was restricted to the primitive ones: union, concatenation and closure. Obviously, the disambiguation semantics can be extended to other, derived operators such as `R?` or `R+` by setting the transition scores accordingly on the choice points introduced by these operators.

The user may also wish to retain some ambiguity, using the parser for filtering rather than finding one optimal solution. In such a case, preference marks are not inserted at the affected nodes of the regular expression. The search algorithm requires a straightforward adaptation to the case of multiple optimal paths.

8 Related Work

Discussion of related work is split into three areas: the grammar formalism, expressing preferences in hand-writable rules and the run-time evaluation of such preference-based grammars.

8.1 Formalism

In the area of text-to-speech synthesis, pioneering finite-state work has been done in the framework of weighted FSTs at Bell Labs (Sproat 1996, Mohri and Sproat 1996). There, as in most finite-state approaches to NLP, text processing tasks are viewed as successive stages of string rewriting, each implemented by an FST; the FSTs for different stages may be combined via FST composition.

In the formalism presented here, a similar idea underlies the *rewrite rules*, formulated in a regular calculus. However, there are also the more structure-oriented *parsing rules*, which establish a tree-shaped derivation for the constructs being normalised. The possibility of producing such derivations has proved to be of great help to the grammar developers.

8.2 Parsing Preferences

There are two basic ways of expressing parsing preferences in a finite-state framework. One is to encode them as real-valued *weights* associated with the transitions of a weighted FST (Mohri 1997). The transducer is ambiguous, but adding weights along alternative paths yields a preference order over the possible parsing results. Thus, the weights may be used to guide the search for the optimal solution. This framework is

[6] The languages covered comprised English, Greek, German, Spanish and French; each of the language-specific grammars consisted of up to one thousand rules.

particularly well-suited for probabilistic NLP approaches where the weights express probabilities (typically as logarithms).

The other way of handling preferences is to apply them at compile time, when particular rules and constraints are combined to form a grammar system. The resulting FST contains only the optimal paths; hence it is unambiguous and can be either determinised or converted to a bimachine.

A possible method of combining prioritised rules is to express each of them as an FST and then join these FSTs via an operation called *priority union* (Karttunen 1998). The priority union of of two FSTs T_1 and T_2 is defined as $T_1 \cup (T_2 \circ (\overline{Upper(T_1)}))$, where $\overline{L}$ is the complement of a regular language L and $Upper(T)$ is the language accepted by transducer T. Priority union restricts transducer T_2 to the complement of T_1. As a result, all conflicts between translations in T_1 and T_2 are resolved in favour of T_1 while strings not accepted by T_1 can still be rewritten by T_2. Similar formalisations have also been given for the longest-match and shortest-match semantics of regular expressions (Karttunen 1996, Gerdemann and van Noord 1999).

The main difficulty related to this "algebraic" approach is that it involves repeated application of costly operations such as regular complement (exponential in $|Q|$) and composition (quadratic in $|Q|$). The resulting FST is non-deterministic, hence another worst-case exponential determinisation step (subset construction or bimachine creation) is required. All this may lead to very slow compilation for realistic text normalisation grammars. In order to estimate the processing overhead caused by these operations, we replaced all instances of regular union in the grammar by priority union. As a result, the compilation time (before determinisation) for the 78-rule grammar evaluated in section 6.3 increased from below 2 seconds to 94 seconds.

One might argue that the running time of these operations is of secondary importance as compilation can be done off-line. However — as already mentioned in section 1 — compilation times exceeding one minute may seriously hamper the productivity of grammar development.

This is the reason why we decided to employ a compilation method that does not eliminate non-optimal paths from the transducer at compile time, but associates its transitions with additional control information (preference markers).[7] From this perspective, our compilation method exhibits a strong resemblance to to the weighted FST paradigm: ambiguity is left in the FST and resolved at runtime using dynamic programming. However, the markers embedded into the output strings of the FST are evaluated in a completely different way than real-valued weights are.

The construction of the FST encoding all the rules is cheap. If N is the number of regular operators and occurrences of symbols in the grammar, it involves a) Thompson construction leading to the creation of an FST with $|Q| = O(N)$ states and $|E| = O(N)$ transitions in time $O(N)$ and b) ϵ-elimination, which can be done in time $O(|Q| \cdot |E|)$. Hence, the construction of the grammar FST is bounded by $O(N^2)$. The observed dependency between N and the actual compilation time is linear. The only costly operation is the creation of the reverse acceptor $\overleftarrow{A}$, which is omitted in the

[7] Still, an unambiguous FST can always be recovered from an FST with preference marks, e.g. using the *directional best paths* algorithm (Eisner 2000, Eisner 2002) designed to implement directional constraint evaluation in the OT framework.

development mode.

8.3 Search for the Optimal Solution

The standard solution to the problem of finding the best path for a string w in an ambiguous FST is to compose the FST with an FSA encoding the input string. If the scores associated with the transitions are viewed as edge weights of a directed graph, finding the best path in the resulting FST is then an instance of the single-source shortest path problem. Viterbi search performs both the composition (creating a structure called a *trellis*) and search for the optimal solution within this structure in time $O(|w|)$.

This method can also be used to find the best-scoring path in the case of our construction. However, parsing would most certainly be less efficient than in the case of a bimachine due to the overhead caused by the construction of the trellis and keeping up to $|Q|$ best path candidates for each string position, which is obviously more expensive than $2 \cdot |w|$ lookups in the transition table (i.e. the total parsing cost in the case of a bimachine).

9 Conclusion

The rule compiler described in this paper presents an alternative to both traditional weighted FST compilation (Mohri and Sproat 1996) and the "algebraic" approach in the vein of Kaplan and Kay (1994). Ambiguity is dealt with using a simple but powerful disambiguation strategy that may be viewed as a mixture of priority union (Karttunen 1998) and longest/shortest match (Gerdemann and van Noord 1999). The use of preference markers makes it possible to avoid the relatively costly operations typically associated with the compilation of the above constructs. On the other hand, the resulting structure (an ambiguous FST containing preference markers and a right-to-left deterministic acceptor) still makes it possible to conduct deterministic search for the optimal result. In this way, the compiler combines fast compilation and efficient processing at runtime.

References

Berstel, J.(1979), *Transductions and Context-Free Languages*, Teubner Verlag.

Eisner, J.(2000), Directional constraint evaluation in Optimality Theory, *Proceedings of COLING 2000*, Saarbrücken, Germany, pp. 257–263.

Eisner, J.(2002), Comprehension and compilation in Optimality Theory, *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, Philadelphia.

Ellison, M.(1994), Phonological derivation in optimality theory, *COLING 1994*.

Gerdemann, D. and van Noord, G.(1999), Transducers from rewrite rules with back-references, *Proceedings of EACL 99*.

Hopcroft, J. E., Motwani, R. and Ullman, J. D.(2001), *Introduction to Automata Theory, Languages, and Computation*, Addison-Wesley.

Hunt, A. and Black, A.(1996), Unit selection in a concatenative speech synthesis system using a large speech database, *Proceedings of ICASSP 96*, Vol. 1, Atlanta, Georgia, pp. 373–376.

Kaplan, R. M. and Kay, M.(1994), Regular model of phonological rule systems, *Computational Linguistics* pp. 331–378.

Karttunen, L.(1996), Directed replacement, *in* A. Joshi and M. Palmer (eds), *Proceedings of the Thirty-Fourth Annual Meeting of the Association for Computational Linguistics*, Morgan Kaufmann Publishers, San Francisco, pp. 108–115.

Karttunen, L.(1998), The proper treatment of optimality in computational phonology, *in* L. Karttunen (ed.), *International Workshop on Finite State Methods in Natural Language Processing*, Association for Computational Linguistics, Somerset, New Jersey, pp. 1–12.

Kempe, A.(2001), Factorization of ambiguous finite-state transducers, *Lecture Notes in Computer Science*.

Mohri, M.(1997), Finite-state transducers in language and speech processing, *Computational Linguistics* **23**(2), 269–311.

Mohri, M. and Sproat, R.(1996), An efficient compiler for weighted rewrite rules, *Meeting of the Association for Computational Linguistics*, pp. 231–238.

Roche, E. and Schabes, Y.(1996), Introduction to finite-state devices in natural language processing, *Technical report*, Mitsubishi Electric Research Laboratories, TR-96-13.

Sproat, R.(1996), Multilingual text analysis for text-to-speech synthesis, *Workshop on Extended Finite State Models of Language (ECAI '96), August 12-16*, von Neumann Society of Computer Science, Budapest, Hungary.

The Interface between Lexical and Discourse Semantics

The case of the light verb "have"

Alexandros Tantos

Universität Konstanz

Abstract

The macro-target of discourse interpretation for computational purposes is the automatic detection of events in a text and their ordering in a temporal scale. Asher and Lascarides' (2003) ideas on the semantics-pragmatics interface between the lexical and discourse level along with the logic mechanism for inferring rhetorical structure (often nonmonotonically) proposes an interesting way of achieving this macro-target. This talk examines the discourse behaviour of the light verb *have* in English, as in "John *had* his students walk out of class." and proposes an extension of Asher and Lascarides' ideas, primarily with respect to the representation of lexical semantics and the interaction with discourse. Light *have* has previously been analyzed from within lexical semantics as a semantically light element which enters into a complex predication with another predicate (e.g., Ritter and Rosen 1993). This talk argues for a different approach to light *have* by taking discourse structure into account. When one looks beyond the domain of lexical semantics, it becomes evident that light *have* supports the inference of discourse relations and functions as a reliable marker for discourse interpretation.

1 The problematic case of the light verb *have*

1.1 The problem

The analysis of light verbs comprises a challenging enterprise in natural language semantics and syntax. The verb *have* in its light use participates in V+V constructions and shows an ambiguity as to the semantic interpretation of its subject in the complex predicate. This paper provides a possible explanation for the variable semantic behaviour of the subject of *have*.

In (1) for example, John is interpreted as the indirect causer of "wash" but in (2) John is interpreted as the indirect undergoer or experiencer of the "overwatering" event. By indirect causer or receiver I mean that the semantics of the subject can not participate in the denotation of the main event. Additionally, although the terms causer and undergoer are used, the interpretation used in this paper is not the same as the one commonly used in the lexical semantic community. The term of "indirect" causation is used here in a much looser sense than commonly accepted in lexical semantics.

(1) John *had* Jim wash his hands.

(2) John *had* Jim overwater the plants in his office.

Things are more complicated when one encounters sentences like (3) taken from Ritter and Rosen (1993). In (3) one cannot interpret the subject of *have* either as agentive or as experiencer, since both of the readings might be possible in different contexts. Ritter and Rosen (1993) argue for a clear case of ambiguity. One can interpret "John" as

either the causer or the experiencer of the "walking out of class" event. In section (3), I claim that this variable behaviour of the subject of *have* reflects the underspecified semantic role of the light verb *have* at the discourse level.

(3) John *had* half his students walk out of his class.

The paper is organized as follows. The first section presents the problematic behaviour of light *have* as it has been identified by previous syntactic and semantic approaches. The second and third sections discuss ideas of Asher and Lascarides (2003) on the lexical-discourse semantics interface and provide the proposed solution in full length. The fourth section focuses on a specific example; under the current analysis, the reliable computation of the discourse meaning is ensured by focusing primarily on the lexical knowledge associated with light *have*, thereby avoiding a heavy use of domain and world knowledge.

1.2 Prior analyses and assumptions

Light verbs participate in monoclausal complex predications. Within LFG, for example, the predicational power of a complex predicate (main predicate plus a light verb) is equal to an f-structure containing only one predicate and a single subject (Alsina 1996, Butt 1995). This monoclausal f-structure is obtained despite the fact that the argument structure is complex and that the light verb often adds an extra argument (cf. the uses of light *have* in (1)–(3)).

However, the syntactic theories have no real answer to the question of how to isolate the precise semantic contribution of the light verb to a complex predication. The problem of identifying the precise semantic contribution of a light verb is illustrated by light *have* as well. Should one assume two independent lexical entries for light *have*, one which contributes an experiencer to the complex predication ((2)) and one which contributes a causer ((1))? Or should one assume an underspecified entry with an underlying semantics that can result in either an experiencer or a causer reading?

Ritter and Rosen (1993) argue convincingly that one should assume only one underlying lexical entry for *have*. I briefly illustrate their basic analysis and propose to build on it by taking the discourse context more seriously than has been done in previous analyses. It is generally recognized that the choice for the differing interpretation of the subject of *have* depends on the surrounding context. However, even though the importance of the discourse context is recognized (see, for example, Levin and Rappaport 1995), previous analyses of light *have* have focused exclusively on the interface between lexical semantics and syntax.

Ritter and Rosen (1993) propose that the argument contributed by light *have* may not be external to a verb but to an event. According to them, the interpretation of the subject of *have* is not dependent upon the lexical semantic content of the main predicate, but upon the role *have* plays in organizing the event it participates in. Thus, there are two ways of attaching and hence interpreting the external argument of *have* to the event denoted by the main predicate: the causer reading is derived from an anchoring of the external argument before the main event; the experiencer reading is

derived via an anchoring of the external argument after the main event. The basic idea is sketched in (4), which provides an analysis of the ambiguity illustrated iin (3).

(4) have&walk out |—— Cause ——|—— Walk out of class ——|

|—— Walk out of class ——|—— Exp. ——|

These analyses are taken from a sentential point of view, but a discoursal perspective onto the same phenomena provides a more useful insight into the reasons of the inherent ambiguous behaviour of the light verb *have*. In order to explore the advantages of integrating lexical and discourse knowledge it is essential to first define the interaction of the two levels of representation. The interface between these two level of description is presented in the next section. It is based on a multi-modular theory of discourse representation and interpretation called Segmented Discourse Representation Theory (SDRT henceforth) introduced by Asher (1993) and further developed by Asher and Lascarides (2003).

2 The interface between the lexicon and the discourse

The representation and inference of textual units has been one of the main focuses of computational semantics since the development of the first dynamic semantic accounts for discourse interpretation by Heim (1982) and Kamp (1983) and later by Kamp and Reyle (1993). It has been applied to discourse processing systems based on the ideas of Hobbs et al. (1993) and other AI techniques. One of the main concerns of those theories is how one can integrate information from totally different direction, which plays a distinct but crucial role to the overall coherent interpretation of the discourse.

After a decade of research in the area, Asher (1993), Asher and Lascarides (2003) tracked the disadvantages of previous approaches and provided a model-theoretic way of interpreting the discourse without excluding crucial information addressed by different knowledge sources, but at the same time maitaining the independence and expressibility required by each of these knowledge sources. The interpretation system is based on a modular inference component called DICE (Discourse in Commonsense Entailment) supported by a modal propositional logic called ”glue” or information packaging logic with a nonmonotonic component. This logic contains default inference axioms for several kinds of discourse or rhetorical relations between two segments and is flexible and “flat” (in terms of using underspecified representations) enough to include the necessary knowledge from lexical and compositional semantics as well as domain, world and cognitive modeling knowledge.

SDRT pays particular attention to the contribution of lexical semantic analyses, since in many cases inference and interpretation can be simplified by lexical information. If domain and world knowledge are heavily used, it becomes extremely difficult to end up with a robust analysis, since pragmatic information is not enough to determine the relations between two or more utterances. This important observation and strategy is particularly beneficial in famous cases as in (5) (Asher and Lascarides, 2003).

(5) Max fell. John pushed him.

The right interpretation of this two-sentence discourse is that there is an *Explanation* relation between the two utterances. Classic analyses would suggest that such an inference is based exclusively on indefeasible explicit world knowledge axioms about pushing events that cause falling ones. However, SDRT uses as much linguistic knowledge as possible. In (5), the verb "push" is transitive and entails movement of the object. The kind of movement is further specified by "fall" in the second utterance. By using lexical information for simplifying the discourse inference, one avoids the use of detailed defeasible domain knowledge axioms for every kind of entailment of pushing events.

SDRT's inference engine integrates linguistic knowledge wherever possible and it seems that both discourse and lexical processing methods may benefit from this strategy. Asher and Lascarides (1995) exploit the possibility of building an interface between the lexicon and the discourse and establish a two-way interaction between the two distinct levels of representation. Lexical processing techniques are enhanced by a lexical semantic representation containing rhetorical relations and, therefore, using information from the discourse for word disambiguation tasks, while the inference of discourse relations has been enriched by valuable lexical semantic information.

SDRT uses a set of discourse relations called rhetorical relations as two place predications between two utterance tokens depending on their connection inside the text. These relations attempt to consider pragmatic effects brought about by the discourse and assign them a model-theoretic interpretation. Some of the rhetorical relations used by the theory for various types of texts and purposes among others are *Narration*, *Explanation*, *Result*, *Elaboration*. The inference of these relations is based on a series of default axioms that can be overridden in favour of the discourse coherence. A pragmatic principle and constraint is called Maximal Discourse Coherence (MDC). This ensures that the discourse update procedure takes pragmatic validity under consideration, chooses the pragmatically most preferable interpretations and excludes the implausible ones. Explicit details about this part of the theory are provided in Asher and Lascarides (2003).

Whatever strategy for whatever reasons one picks up to build the compositional semantic representation of a sentence, it has been proved that lexical and discourse inference should be handled in a delicate manner. This paper adopts some of the strategies of Asher and Lascarides (1995) to use the necessary lexical information for discourse inference. These strategies include:

- The use of underspecified lexical representations in SDRS format in cases of ambiguities. This is unlike the usual methods of word sense disambiguation where one deals with disambiguated lexical items and revises choices.
- The avoidance of lexical disambiguation that could lead to discourse incoherence wherever possible.

3 The light verb *have* and its contribution to discourse interpretation

The variable behaviour of the external argument of *have* and the difficulty of pinning down lexical or syntactic factors determining the integration of the external argument,

suggest that its lexical semantic properties should be established with respect to the larger discourse context. Following Tantos (2004), in (6) one can isolate the role of the light verb *have* only if one examines it in a larger context.

(6) (π_1)John was shouting all the time. He (π_2)*had* a student walk out of the class. (π_3) He felt very bad about it.

Extending the picture argued for by Ritter and Rosen (1993) it becomes evident that the role of *have* is not only to organize the event of the main predicate but also to structure the discourse. The second utterance in the discourse describes the indirect causation relation built between the subject of *have* and the event of the main predicate in the causative reading. Therefore, (π_2) builds a causal connection with (π_1) and by using the default inference axiom of *Result* in (7) provided by SDRT, we get the desired intuitive interpretation that the "shouting" resulted in "a student walking out of the class".

(7) (?(α,β,λ) & *Top*(σ) & *causeD*[1](σ,α,β) & *Aspect*(α,β))>*Result*(α,β,λ).

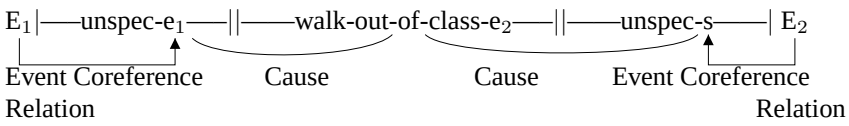
(8) (?(α,β,λ) & *Top*(σ) & *causeD*(σ,β,α) & *Aspect*(α,β))>*Explanation*(β,α,λ).

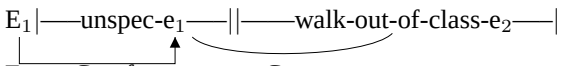
The default axiom in (7) says that if two utterances α and β are to be connected with a rhetorical relation, their top node in the discourse is σ and there is evidence from compositional or/and lexical semantics that there is causal relation between them, independently of their aspect, then one should defeasibly infer a *Result* connection between α and β.

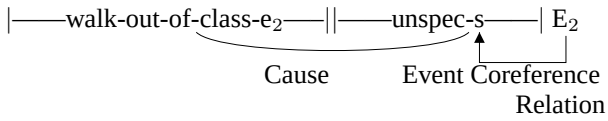
On the other hand, the same causal effect is responsible for the experiencer reading of the subject of *have*, which obtains simultaneously with the causative reading in (7). The single event of the complex predicate in (π_2) causes the effect depicted in the next utterance (π_3). The relevant rhetorical relation to be inferred should be *Explanation*, which is the dual relation to *Result* as defined by Asher and Lascarides' (2003) inference axiom (9), where the interpretation of the temporal order of the events is inverted. At this point, it is worth noting the parallelism to Ritter and Rosen's (1993) analysis, who view the two interpretations of the subject of *have* in terms of different attachment possibilities to the event of the main predicate. However, their analysis concentrates on the sentential level and on the interaction between lexical semantics and syntax.

I take this to suggest an underspecified representation at the discourse level for the light verb *have*. This underdetermines which of the two readings is preferable and is expressive enough to cover the causal effect associated with light *have* in both the experiencer and causer readings. In (9) both possibilities of interpretation are illustrated. This reflects the different temporal placements of the event of the main predicate in an unspecified eventuality.

[1]causeD is a term in SDRT defining the "discourse premissible cause". For more details on the term see Asher and Lascarides (2003).

(9) 

(10) 

(11) 

Further details about the different representations illustrated above and the information they contain follow in the next sections. But first an explanation of certain conventions in the above representations is necessary.

In (10) and (11) the causal link associates the single event of the complex predicate (e_2) with an unspecified eventuality (either the eventive e_1 in the causative reading represented in (10) or the stative s in the experiencer one (11)). This in turn is identified with an external eventuality denoted by the context (E_1 and E_2 for (10) and (11) respectively). This tactic is necessary since it is possible that the cause or the effect of this single event is not expressed at all, as in (12).

(12) John *had* me push him strongly in the back. Nothing happened. The pen was still in his neck.

In (12) the indirect causation is not explicitly mentioned in the text. However, human interpreters can infer that "John" did something to "me" so that i push him in the back. Although what "John" did is not explicitly mentioned, the reader does not face any problem in the interpretation. If the text clarifies it later it is perfectly acceptable; if it is not clarified, the reader can still move on by keeping in mind that "John" did something whatever that might be. Furthermore, the implicit inference about the unspecified eventuality invoked by *have* has to do with the fact brought by the third sentence that the pen was in "John's" neck and this explains why "John" *had* "me" push him. Therefore, we need to allow for the presence of an unspecified eventuality that can or can not be identified with another eventuality in the context.

As mentioned previously, the ambiguous nature of the light verb *have* can be captured by an underspecified representation. Any of the currently available formalisms of underspecification could be used, since the main idea is that the disambiguation process follows the building of a shallow representation by satisfying constraints on tree structure logic formulas. The lexical representation in Figure 1, built in terms of an SDRS, leaves uninstantiated the argument slots in the causeD predication.

The '?' denotes underspecified information (see Asher and Pustejovsky 2000, Asher and Lascarides 2003) that can be specified as part of the discourse analysis, which in turn relies on linguistic clues within the *have*-clause and the surrounding discourse as much as possible (see discussion below). As can be seen in Figure 1,

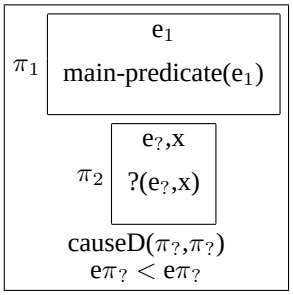


Figure 1: The lexical representation of *have*

the lexical entry for *have* specifies that there must be another eventuality in the clause (labeled π_1). This stands for the verb combining with have, called "main-predicate" in Figure 1 for ease of exposition. We also postulate that part of the semantic contribution of light have is an unspecified eventuality ($e_?$). Although this unspecified eventuality is introduced by *have*, it does not denote *have* itself. Rather, it is a pointer to the unspecified eventuality just mentioned above. Also, the question mark outside ($e_?$,x) is supposed to be predicated of the eventuality and "x" and serves as a place holder for an unknown predicate that will resolve the anaphora in the discourse level.

3.1 Inference and interpretation with light *have*

Asher and Lascarides (1995) use HPSG (Head Phrase Structure Grammar) to establish the connection between a lexical syntactic description and a DRS-like lexical representation. They used the rich type inventory provided by HPSG grammars and the useful tools of reentrancy and unification to drive the semantic composition based on the syntactic description of the properties of the lexical item. These ideas may be applied to any other unification-like formalism, although the different architecture choices should lead to different kinds of analyses.

This paper assumes an LFG (Lexical Functional Grammar) based syntactic representation. LFG is a grammatical theory that proposes a modular projection architecture for the distinct levels of analysis, where each level of description can use individual methods of representation according to its needs and purposes. In that respect LFG resembles SDRT, since both do not provide a derivative model of description but build interfaces —in terms of SDRT— or correspondence functions — in terms of LFG. The syntactic analysis of the *have* sentence from (6), repeated below in (13), is expressed by the f-structure in (14).

(13) (π_1)John was shouting all the time. He (π_2) ($\pi_{2'}$)*had* a student walk out of the class. (π_3) He felt very bad about it.

(14)
$$\begin{bmatrix} \text{PRED} & \text{'have<(}\uparrow \text{SUBJ) 'walk-out<(}\uparrow \text{OBJ}_\theta\text{), (}\uparrow \text{OBL)>'>'} \\ \text{TNS-ASP} & \begin{bmatrix} \text{TENSE} & \text{PAST} \end{bmatrix} \\ \text{SUBJ} & \begin{bmatrix} \text{PRED} & \text{'PRO'} \\ \dots & \end{bmatrix} \\ \text{OBJ}_\theta & \begin{bmatrix} \text{PRED} & \text{'student'} \\ \dots & \end{bmatrix} \\ \text{OBL} & \begin{bmatrix} \text{PRED} & \text{'class'} \\ \dots & \end{bmatrix} \\ \text{VTYPE} & \text{LIGHT-HAVE} \end{bmatrix}$$

Following the analysis of complex predicates proposed by Butt et al. (2003a) and Butt et al. (2003b), the main PRED of the clause is a complex predicate and it has been put together in the syntax via the restriction operator. The complex argument structure is represented as part of the complex PRED and shows that the light verb *have* is responsible for contributing an extra argument to the predication.[2] This extra argument functions as the subject of the complex predicate. The agent of the main event "walk" is realized as the object of the complex predicate and the feature VTYPE is the one that triggers the underspecified lexical representation assumed in Figure 1.

LFG assumes that the f-structure provides those bits of syntactic information that are relevant for further semantic analysis (or machine translation). This assumption is also made in this paper, and as is also standardly assumed, the mapping to semantics is done via the correspondence function σ, which maps f-structures to semantic formulas of any kind of compositional semantic framework. In particular, this paper assumes the ideas presented in Dalrymple (1999) on building compositional semantics based on a resource sensitive procedure driven by linear logic deduction.

The inference of the right relations is based on the main ideas of DICE and is enriched by the contribution of syntactic information at f-structure. The transformation of f-structures into wffs of the glue logic used by DICE is done by using the Attribute Value Logic (AVL) of Johnson (1988). The "translation" from feature descriptions into propositional modal formulas is done in two steps. First, comes the satisfaction of the attribute-value descriptions in the sentential level in terms of Johnson's (1988) AVL. Johnson (1988)[3] has studied the use of a first order quantifier free logic to describe the constraints imposed by lexical items and syntactic rules in order to get the final attribute-value structure at the top syntactic (S or IP) level. In terms of his system, attribute-value structures play only one role: they are defined in order to give a semantics for the language that describes them. Furthermore, following his steps and the original distinction proposed by Kaplan and Bresnan (1982), this paper adopts the distinction between description of a feature structure and the structure itself. In that

[2]For reasons of space, this f-structure has been abbreviated to show "preds-only". For a fuller discussion of f-structures and the grammar development platform XLE, see Butt et al. (1999).

[3]Blackburn (1991) has also worked on defining the semantics of feature structures in a modal logic framework, but his focus is on transferring typed feature structures something which this work does not adopt.

way, one can operate on descriptions of attribute-value structures in a flexible way.[4]

3.2 The causative reading

The subject of light *have* is disambiguated through the context. Although in (13) there is no surface clue that calls for the existence of causation, (π_2) builds a causal connection with (π_1) by considering the lexical information provided by *have*. The immediate inference axiom for disambiguation towards a general causative reading, triggered by the verb *have*, is illustrated in the monotonic indefeasible implication of (15).

(15) have(x,eβ) $\rightarrow$ causeD(?α,β)

This axiom contains mixed information from the lexical and discourse level. It states that if we encounter *have* with its subject argument "x" and an utterance token β that includes the event "e", then one can monotonically infer that there is a causal connection "causeD" and as a second one an event 'eβ', then a causal connection can be made between a proposition token'?α' that includes the unspecified causing event and a second one 'β' that belongs to the syntactic configuration of *have*, namely the proposition token that includes the main predicate.

In order to get the right interpretation in (13) I propose to use and to apply the above inference axiom. In this direction, one needs to proceed in building the event coreference relation between the unspecified eventuality implied by "have' and another event from the context with techniques introduced by Danlos (2000). Briefly, Danlos' (2000, 2003) analysis departs from the commonly accepted fact that causative verbs should be analysed in terms of a complex predication involving a causing subevent (e_1) which brings about a new state (e_2) (e.g., Hale and Keyser 1993, Higginbotham 2000). Under Danlos' analysis, the causing subevent needs explication that comes from the discourse, and the causative verb creates an anchor between its resulting state (e_2) and the explicative coreferent event, which is obtained by a verb from the previous or following utterance. So, in (16), taken from Danlos (2001), there is a coreference relation between the hitting event of the first clause and the cracking causing subevent of the complex event denoted by the verb 'crack' in the second clause.

(16) Fred hit the glass against the sink. He cracked it.

The coreference relation between the hitting and the cracking event can be computed by implications describing lexico-semantic relations, like hyponymy between the events described by the verbs "hit" and "crack". Taking the parallel case of *have*,

[4]Negation and disjunctions on feature descriptions are allowed. In contrast, unification theories that suggest that attribute-value structures are directly associated with lexical items and syntactic rules are not able to handle disjunction or negation, since it is not conceivable what the disjunction of an attribute-value structure would mean.

one can see that it signals that two utterances are to be connected via indirect causation. The main difference is that the relation between the unspecified eventuality implied by *have* and the "external" one is not computed in terms of lexico-semantic connections, since *have* does not give clues as to the nature of the unspecified causing eventuality. Exactly this fact makes the main difference between purely causative verbs and the light verb *have*, namely that *have* provides weaker indications for the building of the possible event coreference relations. The second difference is important for theoretical considerations, since causative verbs involve two subevents very closely related by the causation relation, while under my assumptions *have* relates two events denoted by different verbs only indirectly. Nevertheless, in this paper we can see that it is feasible to build such connections based on specific constraints that *have* imposes on the discourse as shown below. Tantos (2004) has shown that in real texts that include the light verb *have*, certain constraints must be satisfied by the discourse until a resolution is realized in favour of a causative or experiencer interpretation. The constraints that should be satisfied in favour of the causative reading and instantiate the inference axiom in (15) are given below.

- The subject of *have* is anaphorically bound with an entity introduced in one of the previous or following discourse accessible utterances [5] which is always assigned the role of agent by its verb.
- The eventuality type denoted by the verb whose subject is bound with the subject of *have* is always "event".
- This "external" event should be instantiated before the event denoted by the main predicate.

These theoretical observations need to be encoded in axioms of mixed knowledge from syntax, argument structure and discourse semantic representations. The point of complexity in such kind of constraints is that they not only require that two entities corefer, but that their syntactic function and semantic role in the distinct sentences to which they belong should be identical as well. So, one would have to find a way to resolve the anaphora in a restricted way; namely one has to search between the accessible referents whose syntactic function and semantic role is the same with the constraints brought by *have*.

Formally, the corresponding constraints are represented by the following formulas, which include mixed knowledge from different knowledge sources:

- ($\delta_{\mathrm{c}}((\alpha, \mathrm{SUBJ}_{\mathit{have}})$=PRED[6]) & PRED $\rightsquigarrow$ x) & $\exists$ $\mathrm{PRED}_{\mathrm{i}}$ (($\delta_{\mathrm{a}}(\alpha, \mathrm{SUBJ}_{\mathit{REL}})$ = $\mathrm{PRED}_{\mathrm{i}}$) & ($\delta_{\mathrm{a}}(\alpha, \mathrm{AGENT}_{\mathit{REL}})$ = $\mathrm{PRED}_{\mathrm{i}}$) & $\mathrm{PRED}_{\mathrm{i}} \rightsquigarrow$ y)) & x=y
- Eventuality type of $\mathrm{e}\pi_?$ = event
- $\mathrm{e}\pi_? < \mathrm{e}\pi_2$

[5] Here it is meant between the accessible utterances in the discourse tree.

[6] This is the instantiation of the subject of *have* whether it is a pronoun or a proper name; the same applies to the other PRED.

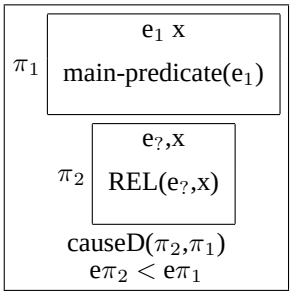


Figure 2: The representation of *have* in the causative reading

The notation of the first constraint needs some clarification, since the current work primarily follows the AVL adopted by Johnson (1988) but has been slightly revised because of current considerations about the availability of lexical information in the discourse level. Briefly, the δ predications in the first constraint denote a function from the domain of attribute value elements to attribute value elements whose first argument inside the brackets is the element which contains the attribute of the second position; e.g. in the first δ function, the α element that contains the SUBJ attribute as one of its elements, is mapped to the attribute PRED. Additionally, functional descriptions in the form of AVL formulas are assigned the subscripts "c" to denote information of the current utterance and "a" to denote information of the discourse accessible utterances. By marking the pieces of information in this way, one can exploit the constraints about availability in the discourse structure and at the same time use the information coming from syntax in terms of functional structure descriptions. Furthermore, "x" and "y" stand for discourse referents and the arrow from attributes to referents represents the mapping from syntactic to semantic representation. Finally, the last conjunct adds the crucial coreference of "x" with "y". In other words, the first constraint says that the subject of *have* is assigned a PRED value [7] and that there exists some PRED_i[8] that belongs to the syntactic representation of the discourse available segments that is assigned the grammatical function SUBJ of an unknown predicate REL, its semantic role is AGENT and is mapped to a referent "y". Also, the two referents should be anaphorically bound in the semantic representation of the discourse.

The second and third constraints encode the relevant conditions about the type of the eventuality that the "external" predicate should denote and about the temporal position of the events denoted by the "external" predicate and the main predicate of the complex predicate respectively.

The second step is trivial and partially follows the algorithm of Asher and Lascarides (2003) for building formulas in their "glue" logic, the main constituent of DICE for updating the discourse. Nevertheless, it is not the subject of this paper to in-

[7] In fact, PRED is an attribute, but it is used here as an variable that is instantiated.

[8] The existential quantification applies over the domain of a restricted set of PREDs that includes the discourse available functional descriptions.

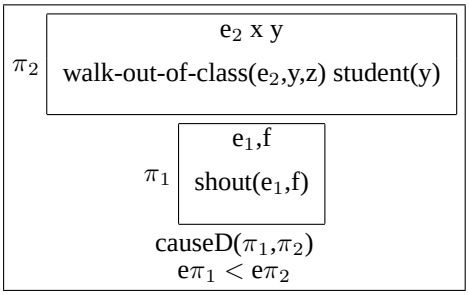


Figure 3: The instantiated lexical-discourse representation of the example

vestigate the properties of such transfer between the logic assumed here for the feature descriptions and the "glue" logic[9]. The above constraints imposed by a lexical item override the limits of the sentence and/or proposition and show that there is a clear interaction between the lexical and discourse level. The event coreference relation between the unspecified event implied by *have* and an external event from the context resolves the underspecification inherent at the lexical representation of *have* and the desired instantiation of the underspecified representation is taking place as illustrated in Figure 2.

The instantiated representation in Figure 3 gives us the desired results. The arguments in the causeD predication have been resolved, the temporal order of the events has been determined and the REL predicate (which designates the unknown external verb in this case) has been instantiated by the relevant predicate. In this way, specific linguistic observations can be exploited by a discourse analysis and lexical items are able to specify a constrained connection with items from the surrounding utterances.

Now let us try to compute the meaning of the small discourse in (13). Inserting lexical knowledge in the discourse level includes the introduction of a new utterance token ($\pi_{2'}$) following the above lexical representation. The resolution of the underspecified entry in favour of one or the other reading depends on which utterance is related with the utterances represented by *have*. The sentence that contains *have* interprets its subject as causative with respect to (π_1) and experiencer with respect to (π_3).

The position of the arguments in the causeD predication is fulfilled if the constraints of one of the two readings are satisfied. In the case of relating (π_2) with (π_1) in (14), the three constraints described above for the interpretation of the causative reading are satisfied. The set of accessible utterances includes only (π_1). The referent in (π_1), mapped to the SUBJ attribute, is identified with the referent mapped to the SUBJ of *have* and is assigned the semantic role of agent. Furthermore, the eventuality type denoted by the verb "shout" is an "event" satisfying the second constraint. Therefore, establishing the temporal relation $<$ that creates a partial order between events

[9] An extended discussion about the relevant algorithm can be found in Asher and Lascarides (2003).

in the discourse, "shouting" $<$ "walking-out-of-the-class" means that the "shouting" event occurred before the "walking-out-of-the-class" event.

The instantiated lexical-discourse interpretation is as in Figure 3[10] and the inference of the right rhetorical connections between the utterances is now feasible. The antecedent of the default axiom for *Result* is satisfied and through the application of a weaker version of modus ponens introduced by DICE, called Defeasible Modus Ponens, one arrives at the intuitively correct interpretation. The "shouting" caused the "walking-out-of-the-class" event.

The interaction between lexical and discourse semantics proposed here, is defined in terms of underspecified lexical entries which can be resolved with the help of discourse information. Therefore, the inference and interpretation of the discourse will resolve the lexical underspecification and provide the necessary interpretation axioms for the lexical item of *have* in the discourse. Tantos (2004) describes the semantic contribution of "have" with the interpretation axiom in (17).

(17) $\text{have}_{\text{causative}}(\text{x},\text{e}\beta) \rightarrow \text{cause}(\text{e}\alpha,\text{e}\beta)$ & $\text{Agent}(\text{e}\alpha,\text{x})$ & $\text{e}\alpha < \text{e}\beta$

This axiom states that if the discourse implies the causative interpretation, then in the model there has to be a causation relation between the two events eα and eβ belonging to the distinct utterance tokens α and β, the subject of *have* should be assigned the role AGENT and the event eα occured before the event eβ.

3.3 The experiencer reading

The inference over and interpretation of the subject of *have* in the experiencer reading are driven by the same strategy and principles. There is also a causal relation between an "external" event and the event denoted by the main verb of the complex predicate. The main difference is that the "external" event is affected by the event of the main predicate of the complex predicate and not the other way around as in the causative interpretation. Therefore, the inference axioms are different. In (13) the "walking out of the class" event is interpreted as affecting the subject of *have* if the sentence is related with the coming one. Thus, the discourse context provides information about the kind of affectedness involved. The underspecified lexical representation of *have* is instantiated so that the arguments in the causeD predication take the right position and the events are placed in the right temporal order.

The inference axiom for the experiencer is similar to the one for the causative reading, but with the inverse order of the arguments, as shown in (18).

(18) $\text{have}(\text{x},\text{e}\beta) \rightarrow \text{causeD}(\beta,?\alpha)$

Again, one needs to build the coreference relation between the unspecified eventuality included in an accessible utterance token $?\alpha$ and the event of the complement in order to apply the above inference axiom. The building of such a coreference relation requires that new constraints need to be satisfied.

[10] I ignore the representation of plurals and the contribution of the prepositions here for reasons of simplicity of illustration

- The subject of *have* is anaphorically bound to an entity introduced in one of the previous or following discourse accessible utterances [11] and is always assigned the role of experiencer by its verb.
- The eventuality type denoted by the verb whose subject is bound to the subject of *have* is always "state", since only stative verbs have "experiencer" subjects.
- This "external" event should be instantiated after the event of the main predicate of the complex predicate, since it is affected by it.

Following the mapping of syntactic information by AVL, the relevant constraints for the satisfaction of the inference axiom (18) are:

- (δ_c((α, SUBJ$_{\textit{have}}$)=PRED) & PRED $\rightsquigarrow$ x) & $\exists$ PRED$_i$ ((δ_a(α, SUBJ$_{\textit{REL}}$) = PRED$_i$) & (δ_a(α, EXPERIENCER$_{\textit{REL}}$) = PRED$_i$) & PRED$_i$ $\rightsquigarrow$ y)) & x=y
- Eventuality type of e$\pi_?$ = state
- eπ_2 $<$ e$\pi_?$

In (13), the interpretation of the third segment in relation to the second one is that its event is causally affected by the main predicate of the second sentence and this is driven by the resolution of the lexical underspecification expressed by *have* in the second sentence. The instantiation of the underspecified lexical entry in the case of the experiencer reading in (13) is as in Figure (4). As in the resolution of the causative lexical representation, one checks whether the above constraints are satisfied. The "external" predicate represented by the predicate "feel" maps its subject to the same referent introduced by *have*, namely "x". The eventuality of π_2 that is denoted by the main predicate "walk out of the class" of the complex predicate affects the eventuality denoted by the "external" predicate "feel" in π_3 and therefore it occurs before it. Additionally, the predicate "feel" assigns the experiencer role to its subject, satisfying all the constraints for the resolution of the lexical entry. The event coreference relation is thus realized successfully and the discourse representation of π_2 and π_3.

4 Conclusion and further thoughts

The analyses for both of the readings of the light verb *have* provide an example of how one can exploit the linguistic information supplied by the context and lexical semantics in order to succeed in the semantic analyses of texts.

The light verb *have* provides an interesting case study for looking at the interaction between lexical semantics and disourse. The current approach brings a new insight into the disambiguation process, since it attempts to integrate a formal theory of discourse interpretation. Also, this paper represents a first step towards building a formal relation between SDRT and LFG, using the AVL proposed by Johnson (1988).

[11] This means that between the available discourse utterances following the principle of accessibility imposed by the discourse structure

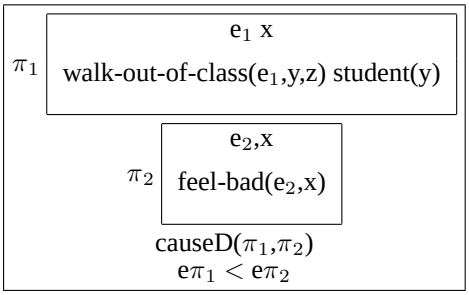


Figure 4: The representation of *have* in the experiencer reading

However, it is by no means implied that these and only these constraints for the resolution of the lexical underspecification apply in order to disambiguate all the cases of the light verb *have*. Domain and world knowledge must be included as a necessary last resort for the disambiguation of the light verb *have* when the above linguistic constraints cannot be met.

References

Alsina, A.(1996), *The Role of Argument structure in Grammar*, CSLI Publications.

Asher, N.(1993), *Reference to Abstract Objects in Discourse*, Kluwer Academic Publishers.

Asher, N. and Lascarides, A.(1995), Lexical disambiguation in a discourse context, *Journal of Semantics* **12**(1), 69–108.

Asher, N. and Lascarides, A.(2003), *Logics of Conversation*, Cambridge University Press.

Blackburn, P.(1991), Modal logic and attribute value structures, *in* M. de Rijke (ed.), *Formal Methods in the Study of Language*, Mathematisch Centrum, pp. 277–322.

Butt, M.(1995), *The Structure of Complex Predicates in Urdu*, CSLI Publications.

Butt, M., King, T. H. and Maxwell III, J. T.(2003a), Complex predicates via restriction, *Proceedings of the LFG03 Conference*.

Butt, M., King, T. H. and Maxwell III, J. T.(2003b), Productive encoding of Urdu complex predicates in the ParGram project, *Proceedings of the Workshop on Computational Linguistics fot the Languages of South Asia Explanding Synergies with Europe*.

Butt, M., Nino, M.-E. and Segond, F.(1999), *A Grammar Writer's Cookbook*, CSLI Publications.

Danlos, L.(2000a), Event coreference between two sentences, *in* H. Bunt and R. Muskens (eds), *Computing Meaning 2*, Kluwer Academic Publishers.

Danlos, L.(2000b), Event coreference in causal discourses, *in* P. Bouillon and F. Busa (eds), *Meaning of Word*, Cambridge University Press.

Hale, K. and Keyser, J.(1993), On argument structure and the lexical representation of syntactic relations, *in* K. Hale and J. Keyser (eds), *The View from Building 20*, MIT Press.

Heim, I.(1982), *The Semantics of Definite and Indefinite Noun Phrases*, PhD thesis, University of Massachussetts.

Higginbotham, J.(2000), On events in linguistic semantics, *in* J. Higginbotham, F. Pianesi and A. C. Varzi (eds), *Speaking of Events*, Oxford University Press.

Hobbs, J. R., Stickel, M. and Martin, P.(1993), Interpretation as abduction, *Artificial Intelligence* **63**(1–2), 69–142.

Johnson, M.(1988), *Attribute Value Logic and theory of Grammar*, Chicago University Press.

Kamp, H.(1981), A theory of truth and semantic representation, *in* M. B. J. Stokhof, J. A. G. Groenendijk and T. M. V. Janssen (eds), *Formal Methods in the Study of Language*, Mathematisch Centrum, pp. 277–322.

Kamp, H. and Reyle, U.(1993), *From Discourse to Logic: Introduction to Modeltheoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory*, Kluwer Academic Publishers.

Kaplan, R. M. and Bresnan, J.(1982), Lexical-Functional Grammar: A formal system for grammatical representation, *in* J. Bresnan (ed.), *The Mental Representation of Grammatical Relations*, The MIT Press, pp. 173–281.

Maxwell, III, J. T. and Kaplan, R. M.(1993), The interface between phrasal and functional constraints, *Computational Linguistics* **19**, 571–589.

Ritter, E. and Rosen, S. T.(1993), The independence of external arguments, *Proceedings of the Twelfth West Coast Conference on Formal Linguistics*, pp. 1–11.

Tantos, A.(2004), The light verb “have” as a discourse active element, *Proceedings of the Sixth International Workshop on Computational Semantics*, pp. 413–416.

List of contributors

Tamás Biró
Alfa-Informatica, Rijksuniversiteit Groningen
P.O. Box 716, NL-9700 AS Groningen (NL)
birot@let.rug.nl

Dorothee Beermann
Department of Linguistics
NTNU, N-7491 Trondheim (N)
dorothee.beermann@hf.ntnu.no

Wauter Bosma
University of Twente, Dept. of Human Media Interaction
P.O. Box 217, NL-7500 AE Enschede (NL)
bosmaw@cs.utwente.nl

Gosse Bouma
Alfa-Informatica, Rijksuniversiteit Groningen
P.O. Box 716, NL-9700 AS Groningen (NL)
gosse@let.rug.nl

Anne Hendrik Buist
Alfa-Informatica, Rijksuniversiteit Groningen
P.O. Box 716, NL-9700 AS Groningen (NL)
ahbuist@gmail.com

Christiano Chesi
University of Siena/MIT
Centro Interdipartimentale di Studi Cognitivi sul Linguaggio
P.zza S. Francesco, 8, I-53100 Siena (I)
chesi@media.unisi.it

Tim van de Cruys
Alfa-Informatica, Rijksuniversiteit Groningen
P.O. Box 716, NL-9700 AS Groningen (NL)
t.v.d.cruys@rug.nl

Walter Daelemans
CNTS-Language Technology Group
University of Antwerp - CDE, Universiteitsplein 1, B-2610 Wilrijk (B)
walter.daelemans@ua.ac.be

Nicole Grégoire
UIL/OTS
University of Utrecht, Trans 10, NL-3512 JK Utrecht (NL)
nicole.gregoire@let.uu.nl

Jon Atle Gulla
Department of Computer and Information Science
NTNU - Gloshaugen, N-7491 Trondheim (N)
jon.atle.gulla@idi.ntnu.no

Lars Hellan
Department of Linguistics
NTNU, N-7491 Trondheim (N)
lars.hellan@hf.ntnu.no

Véronique Hoste
CNTS-Language Technology Group
University of Antwerp - CDE, Universiteitsplein 1, B-2610 Wilrijk (B)
veronique.hoste@ua.ac.be

Wessel Kraaij
TNO TPD
P.O. Box 155, NL-2600 AD Delft (NL)
kraaij@tpd.tno.nl

Shalom Lappin
Department of Philosophy
King's College London, The Strand, London WC2R 2LS (UK)
shalom.lappin@kcl.ac.uk

Kim Luyckx
CNTS-Language Technology Group
University of Antwerp - CDE, Universiteitsplein 1, B-2610 Wilrijk (B)
kim.luyckx@ua.ac.be

Jori Mur
Alfa-Informatica, Rijksuniversiteit Groningen
P.O. Box 716, NL-9700 AS Groningen (NL)
j.mur@rug.nl

Lonneke van der Plas
Alfa-Informatica, Rijksuniversiteit Groningen
P.O. Box 716, NL-9700 AS Groningen (NL)
vdplas@let.rug.nl

Michaela Poß
Leiden University Centre for Linguistics
P.O. Box 9515, NL-2300 RA Leiden (NL)
m.poss@let.leidenuniv.nl

Atle Prange
Businesscape
P.O. Box 1273, N-7462 Trondheim (N)
atle.prange@businesscape.no

Stephan Raaijmakers
TNO TPD
P.O. Box 155, NL-2600 AD Delft (NL)
raaijmakers@tpd.tno.nl

Wojciech Skut
Rhetorical Systems
4 Crichton's Close, Edinburgh EH8 8DT, Scotland
wojciech@rhetorical.com

Alexandros Tantos
Computational Linguistics, University of Konstanz (D)
alexandros.tantos@uni-konstanz.de

Frank van Eynde
Centrum voor Computerlinguïstiek, KU Leuven
Maria-Theresiastraat 21, B-3000 Leuven (B)
frank.vaneynde@ccl.kuleuven.be

Ton van der Wouden
Leiden University Centre for Linguistics
P.O. Box 9515, NL-2300 RA Leiden (NL)
t.van.der.wouden@let.leidenuniv.nl