GAMBIT

-

Genetic Algorithm for Model-Based
mixed-Integer opTimization

Netherlands Organisation for Scientific Research

This research was supported by the Netherlands Organisation for Scientific Research (NWO)

SIKS Dissertation Series

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

ISBN: 978-94-028-2132-1

GAMBIT

-

Genetic Algorithm for Model-Based mixed-Integer opTimization

Genetisch Algoritme voor Model-geBaseerde gemengd-geheeltallIge opTimalisatie

(met een samenvatting in het Nederlands)

Proefschrift

ter verkrijging van de graad van doctor aan de
Universiteit Utrecht
op gezag van de
rector magnificus, prof.dr. H.R.B.M. Kummeling,
ingevolge het besluit van het college voor promoties
in het openbaar te verdedigen op

woensdag 26 augustus 2020 des middags te 12.45 uur

door

Krzysztof Leszek Sadowski

geboren op 29 mei 1985
te Bielsko-Biala, Polen

Promotoren: Prof. dr. P.A.N. Bosman
Prof. dr. M.M. Dastani
Copromotor: Dr. ir. D. Thierens

Dit proefschrift werd (mede) mogelijk gemaakt met financiële steun van de Nederlandse Organisatie voor Wetenschappelijk Onderzoek (NWO)

Contents

1 Introduction

Optimization is a broad concept that can be loosely defined as increasing the effectiveness in solving problems we are faced with. As such, optimization is widely used across many disciplines. In computer science, optimization often refers to improving the performance of a given algorithm through the design and implementation of new and more efficient procedures. In addition, there are optimization algorithms, which typically refers to automatically finding (near-)optimal solutions to optimization problems, typically defined in terms of objective functions to be maximized or minimized as a function of a set of variables.

Because of the widespread application of optimization algorithms, many prominent fields of research that focus on or heavily rely on optimization have been established and actively continue to expand and grow. Optimization problems and algorithms are frequently categorized at the highest level according to the types of variables that are involved. Two commonly encountered types are discrete and continuous optimization. The focus of discrete optimization is finding of optimal solutions for problems of which the solutions are countable. The problem variables are then represented by integer, binary or categorical values. In continuous optimization, problem variables are continuous and are most often represented by real-valued variables. While discrete and continuous optimization are often treated disjointedly, one can easily imagine a problem where both discrete and continuous variables are present simultaneously. Such problems are known as mixed-integer optimization problems, and while common in the real-world practice, they are often very challenging to solve [23] [21] [40].

Key to defining effective and efficient optimization algorithms is exploiting problem structure and other problem-specific information. Sometimes, such problem-specific knowledge is well-defined and known a-priori. This is referred to as white-box optimization. While a white-box setting approach can be very efficient, algorithms exploiting problem-specific information are very often highly-tailored to deal with only the problem at hand making it difficult to generalize them to other problems. Additionally, problem information may be unknown, incomplete, or incomprehensible prior to optimization. This is referred to as the black-box setting. In

order to exploit the structure of problems in a black-box setting, this information needs to be extracted, learned and processed during execution. While more challenging, approaching a problem in the black-box setting allows for the creation of more generic methodologies that can be applied to variety of problems and does not require the understanding or specification of the problem structure a-priori. Such problems are common in many real-world settings, making black-box algorithms potentially very powerful and desirable.

In this thesis, we design a methodology for solving mixed-integer problems in the black-box setting. In order to accomplish this, we take an in-depth look at some existing algorithmic solutions and consider their applicability to the mixed integer domain. We specifically turn to the field of model-based Evolutionary Algorithms (EAs). EAs are optimization algorithms that are loosely based on general concepts of natural evolution. These algorithms, pertaining to the research field known as evolutionary computation, iteratively generate better solutions to a problem through interleaving the selection of better solutions and creating variations of solutions that are maintained in a population. Competent EAs exist in both, discrete and continuous optimization domains, achieve efficient and scalable results on many well-known benchmark problems. In this thesis we aim to see if the abilities of such EAs in their individual fields can be retained when expanding the scope to mixed-integer optimization.

Furthermore, much optimization literature focuses on solving a single, potentially constrained, objective. This often does not represent problems in the real world, where multiple objectives are often present that need to be optimized simultaneously. In many such cases, different objectives may directly oppose each other, introducing an additional layer of difficulty. As EAs are well-suited for multi-objective optimization in a black-box setting, this further motivates our decision to study them in our work.

The remainder of this chapter is organized as follows. In Section 1.1 we formally define mixed-integer problems, and summarize their importance and associated challenges. In Section 1.2 we then provide an overview of the key research objectives of this thesis, as well as a summary of the main results and observations. Additionally, an overview of peer-reviewed publications which provide the backbone of this thesis is provided.

1.1 Mixed-Integer Optimization

Mixed-integer optimization focuses on solving problems where discrete and continuous variables exist simultaneously in the problem definition. This means that optimization challenges that arise in both, discrete and continuous optimization are all potentially present when dealing with a mixed-integer problem. Additionally, because of the existence the different variable types, cross-domain interactions between the discrete and continuous variables may also exist, and can pose addi-

tional challenges. Such cross-domain interactions, caused by variable constraints or objective-function-specific variable dependencies, in combination with already existing challenges inherited from the discrete and continuous domains make mixed-integer problems generally particularly difficult to solve. In this thesis we refer to such cross-domain dependencies as inter-variable dependencies, while variable dependencies exclusive to the same domain type, discrete or continuous, are referred to as intra-variable dependencies.

A Mixed-Integer Problem is defined as follows:

$$min \;\; f(\boldsymbol{x}) = f(\boldsymbol{x_d}, \boldsymbol{x_c}) \tag{1.1}$$

$$s.t. \;\; h(\boldsymbol{x}) = \mathbf{h}(\boldsymbol{x_d}, \boldsymbol{x_c}) = 0, \;\; g(\boldsymbol{x}) = \mathbf{g}(\boldsymbol{x_d}, \boldsymbol{x_c}) \leq 0 \tag{1.2}$$

where $\boldsymbol{x}$ represents the solution

$$\boldsymbol{x} = \boldsymbol{x_d}\boldsymbol{x_c} = d_0 ... d_{l_d - 1} \; c_0 ... c_{l_c - 1} \tag{1.3}$$

where $d_i \in D \in \mathbb{N}, c_i \in C \in \mathbb{R}$ and $\boldsymbol{x_d}$, $\boldsymbol{x_c}$ are the groups of all discrete and continuous variables, respectively. In this thesis, although it does not affect generality, $D \in \{0, 1\}$. f is the objective function (potentially a vector for multi-objective version). **h** and **g** are sets of equality and inequality constraint functions respectively. If both of these sets are empty, the Mixed-Integer problem is said to be unconstrained.

Mixed-Integer problems can be categorized as either linear programming (LP) or non-linear programming (NLP) problems [23] [5]. Key characteristics of a linear programming problem is that the objective function, as well as the equality and non-equality constraint functions are linear in x. An important consequence of this is that the feasible solution space of LP problems is convex, making such optimization problems relatively simple. Many mixed-integer LP (MILP) solvers directly exploit and rely on these characteristics [7].

In a real-world setting, such assumptions may not hold. Mixed-integer problems that do not guarantee the linearity of the objective or constraint functions, and therefore do not guarantee the convexity of the solution space are known as mixed-integer non-linear programming (MINLP) problems [39] [81] [2]. Throughout this work we make no assumptions about the nature of the objective and constraint functions, resulting in a generally challenging mixed-integer non-linear programming (MINLP) problem setting.

1.2 Research Objectives and Thesis Overview

In this thesis we aim to design a novel model-based EA that is capable of achieving efficiently scalable optimization for case of mixed-integer optimization under the assumption that the optimization problem has exploitable problem structure (of non-trivial nature). To this end, we aim to leverage already existing and proven to be highly-competent EA mechanisms for discrete and continuous optimization, as well as introduce new mechanisms catered specifically to the intricacies inherent to the mixed-integer domain. To test our approach we will consider a set of mixed-integer

problems with differently challenging characteristics, and compare our algorithm with a selection of other state-of-the-art algorithms. Finally, we aim to achieve results for a real-world application and showcase a proof-of-concept for a cancer treatment pre-planning optimization.

Provided below is a general overview of the research objectives and main results in the remainder of this thesis as well as citations of first-author peer-reviewed publications related to different chapters of this thesis.

Chapter 2

This chapter provides an overview of the key foundations of evolutionary algorithms that are of importance to the research in this thesis, and introduces the reader to some core terminology and challenges associated with evolutionary computation. Model-based evolutionary algorithms are introduced and explained in more detail. Particularly, two existing state-of-the-art model-based evolutionary algorithms are introduced (LTGA and iAMaLGaM), which have achieved very good optimization results in the domains of discrete and continuous optimization respectively. Some of their main components provide the foundation for GAMBIT, the algorithm that the research in this thesis ultimately has led to.

Chapter 3

Before looking closer at mixed-integer problem optimization, we first examine the flexibility and capabilities of discrete model-based evolutionary algorithms, in particular the LTGA which governs the discrete part in GAMBIT, by considering a well-known complex optimization problem: maximum satisfiability (Max-Sat). Focused on discrete optimization, we consider different optimization approaches and local search hybrids in a black- and white-box settings to determine how known or learned problem structure processing can be useful in solving max-sat problems. A flexible family-of-subsets structure is used as well as mutual information based dependency learning with LTGA, which becomes a permanent feature of the GAMBIT algorithm developed in further chapters. This is also the cause for the acceptance of solutions of equal quality when LTGA performs variation, which is of prime importance especially in the presence of large plateaus in the optimization landscape.

This chapter is based on the following publication:

K. L. Sadowski, P. A. N. Bosman, and D. Thierens. On the Usefulness of Linkage Processing for Solving MAX-SAT. In Genetic and Evolutionary Computation Conference, GECCO '13, Amsterdam, The Netherlands, July 6–10, 2013, pages 853–-860, 2013.

Chapter 4

One of the key challenges of mixed-integer optimization is the potential occurrence of various types of dependency between problem variables. In this chapter we identify subproblems of different levels of complexities that impose different types of such variable dependencies for both discrete and continuous optimization. These

are then used to introduce new mixed-integer problems which vary in the composition of variable dependency types and variable proportions from each domain and introduces new sub functions with interdependent variables of both the discrete and continuous type [78]. This chapter provides motivation as to why the use of problems designed with these fundamental components is a useful and a powerful way of examining performance of mixed-integer problems.

This chapter is based on the following publication:

K. L. Sadowski, D. Thierens, and P. A. N. Bosman. Combining Model-based EAs for Mixed-Integer Problems. In Parallel Problem Solving from Nature – PPSN XIII, pages 342-–351, 2014.

Chapter 5

In this chapter we introduce GAMBIT: a Genetic Algorithm for Model-Based mixed-Integer opTimization. We designed GAMBIT by integrating mechanisms from well-known discrete and continuous model-based EAs. In addition, novel mechanisms that allow explicit learning and processing of mixed-variable dependencies inherent to mixed-integer problems are proposed. Thanks to a balanced integrated models mechanism, a clustering algorithm and an adaptation of a parameter-free scheme, GAMBIT is a parameter-free out-of-the-box optimization algorithm for the mixed-integer domain. Through empirical experiments we show how each of the components included in GAMBIT improves the performance of the algorithm. In addition, we consider a selection of alternative algorithms and highlight the strengths and benefits of using GAMBIT, as well as potential drawbacks. We observe that GAMBIT is a good, scalable alternative for mixed-integer optimization.

This chapter is based on following publications:

K. L. Sadowski, D. Thierens, and P. A. N. Bosman. GAMBIT: A Parameterless Model-Based Evolutionary Algorithm for Mixed-Integer Problems. Evolutionary Computation, 26(1):117—143, 2018.

K. L. Sadowski, P. Bosman, and D. Thierens. Learning and Exploiting Mixed Variable Dependencies with a Model-Based EA. In IEEE Congress on Evolutionary Computation (CEC'16), pages 4382—4389, 2016.

K. L. Sadowski, P. A. Bosman, and D. Thierens. A Clustering-Based Model-Building EA for Optimization Problems with Binary and Real-Valued Variables. In Proceedings of the 2015 on Genetic and Evolutionary Computation Conference, GECCO'15, pages 911—918, New York, NY, USA, 2015. ACM.

Chapter 6

In this chapter we introduce a multi-objective version of GAMBIT, referred to as MO-GAMBIT. Most multi-objective mixed-integer optimization algorithms handle

multiple objectives through some form of objective scalarization. With the addition of dedicated multi-objective mechanisms to GAMBIT, a novel multi-objective evolutionary algorithm for the mixed-integer domain is obtained that can find a complete trade-off set of solutions that approximates the optimal, i.e. Pareto, front. We show that MO-GAMBIT outperforms a restart version of single-objective GAMBIT in which different weighted scalarizations are used as well as a selection of other multi-objective optimization algorithms for mixed-integer problems. We conclude that MO-GAMBIT provides a useful foundation for real-world, multi-objective mixed-integer optimization.

This chapter is based on the following publication:

K. L. Sadowski, D. Thierens, and P.A.N. Bosman. Optimization of Multi-Objective Mixed-Integer Problems with a Model-Based Evolutionary Algorithm in a Black-Box Setting. *[Submitted for review (journal article))]*

Chapter 7

In this chapter MO-GAMBIT is applied to a real-world medical application: automatically optimizing brachytherapy prostate cancer treatment plans. Brachytherapy is a form of radiotherapy whereby a radiation source is guided near tumors, using devices such as catheter implants. In the present clinical work-flow, catheters are first placed inside or close to the tumor based on clinical expertise. Subsequently, software is used to design a plan for the delivery of radiation. Treatment planning is essentially a multi-objective optimization problem, where conflicting objectives represent radiation delivered to tumor cells and healthy cells. However, current clinical software collapses this information into a single-objective, constrained optimization problem. Moreover, catheter positioning is typically not included. As a consequence, it is hard to obtain insight into the true nature of the trade-offs between key planning objectives and the placement and number of required catheters. Such insights are however crucial in understanding how better treatment plans may be constructed. To obtain such insights, we interface with real-world clinical software and derive potential catheter positions for previously treated patients. Selecting and configuring catheters while obtaining insight into the most important trade-offs requires multi-objective mixed-integer optimization. To this end, we use MO-GAMBIT. Our results indicate that clinically acceptable plans of high quality may be achievable with less catheters than typically used in current clinical practice [80].

This chapter is based on the following publication:

K. L. Sadowski, M. C. van der Meer, N. H. Luong, T. Alderliesten, D. Thierens, R. van der Laarse, Y. Niatsetski, A. Bel, and P. A. N. Bosman. Exploring Tradeoffs Between Target Coverage, Healthy Tissue Sparing, and the Placement of Catheters in HDR Brachytherapy for Prostate Cancer Using a Novel Multiobjective Model-based Mixed-integer Evolutionary Algorithm. In Proceedings of the Genetic and Evolutionary Computation Conference, GECCO '17, pages 1224-–1231, New York, NY, USA, 2017. ACM.

Chapter 8

In this final chapter we provide a short overview and summary of our work. We furthermore discuss limitations of our approach as well as potential extensions to further improve our results.

2 Evolutionary Algorithms

Evolutionary Algorithms (EAs) can be summarized as meta-heuristic population-based optimization algorithms. EAs are the algorithms found in the field of Evolutionary Computation which is a subfield of artificial intelligence. EAs are loosely based on the Darwinian concepts of biological evolution, in which a population of solutions improves over time, utilizing the improvements discovered in past iterations [36] [26] [34]. EAs have been shown to be a powerful approach to solving many types of optimization problems in case of discrete as well as continuous problem variables. While many different variants of EAs exist across different domains, a common underlying idea and general functionality is shared between them.

This chapter describes the core, shared, concepts present across different EA-based optimization approaches. These core concepts are explained by means of taking the simple Genetic Algorithm (sGA) as an example, as this relatively simple EA for discrete representations can be considered to encompass many characteristics of evolutionary-based optimization [45][46]. Pros and cons of this approach are considered, and an introduction is made to model-based EAs, which address some of the known shortcomings of simpler EAs.

Two well-known model-based EAs, the Linkage Tree Genetic Algorithm (LTGA) and the incremental Adapted Maximum-Likelihood Gaussian Model Iterated Density Estimation Evolutionary Algorithm (iAMaLGaM), are also introduced in detail in this chapter, as they provide the backbone to the mixed discrete and continuous optimization capabilities of GAMBIT, explained in further chapters of this thesis.

2.1 Basic Terminology and simple Genetic Algorithm

The Simple Genetic Algorithm (sGA) is not directly used to design GAMBIT. However, it provides a foundation of population-based optimization approaches for discrete representations. While evolutionary-based optimization has substantially evolved since the introduction of sGA, the core concepts associated with structures and terminology are still very much present. Moreover, sGA provides a good example of a simple, yet effective approach.

A solution encoding (also called solution) s with a discrete representation consists of a set of **variables** $d_0, .., d_{l-1}$, where d_j represents a value taken from a discrete

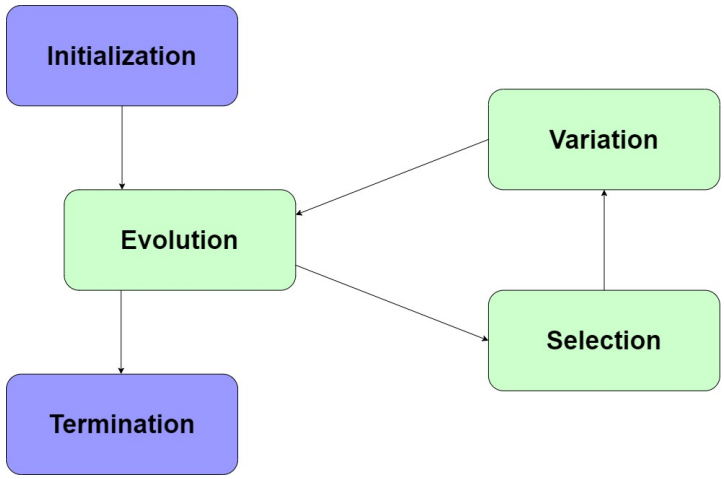


Figure 2.1: *A simple visualization of the evolutionary algorithm framework*

set of possibilities for a specific problem variable and l is the number of problem variables. Such a solution, when assessed with an evaluation function (also called fitness function), results in a **fitness value** that represents how well this solution solves the given problem. In discrete optimization, all the problem variables are discrete: either binary, or integer. A set of such solutions $s = s_1, .., s_{n-1}$, forms a **population**, where s_i denotes the i-th solution in the population with population size n, as illustrated in Figure 2.2.

A population of solutions is iteratively changed (i.e., evolved). Each iteration is called a **generation**. The goal is to improve the population over time with regards to the fitness function. This, in any EA, is accomplished by two mechanisms: **selection and variation**. While the specifics of these mechanisms can vary substantially across different algorithms, the main idea remains the same. Better solutions are selected (more frequently) and used to create new solutions by making variations of these solutions, making use of the joint information that they represent, e.g., by making combinations of the selected solutions. In sGA, selection and variation are clearly identifiable phases, or operators, in the algorithm. Selection creates a set of so-called **parent** solutions that are subject to the variation operations, resulting in evolved, so-called **offspring** solutions. A new population of solutions is constructed by means of combination of the offspring and the old population, which could entail entirely discarding the old population. The process then begins anew. In practice, an EA often starts with a randomly initialized set of solutions and the generational advancement continues until some **termination criterion** is met. The general outline of an EA is illustrated in Figure 2.1.

In the sGA, the selection phase often utilizes so-called tournament selection, where randomly selected small sets of solutions from the population compete against each other, in terms of their fitness value, and only the best solution is copied to the parent pool. This results in only high fitness solutions reaching the variation stage, where they are processed further to make offspring solutions. In sGA, the variation

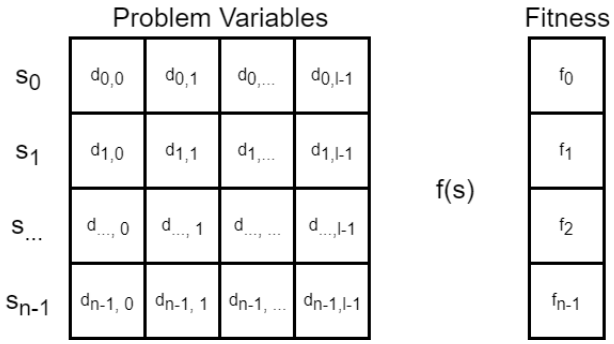


Figure 2.2: *Population of a genetic algorithm, with population size n and number of problem variables l, and fitness evaluation function f.*

stage generally includes two operations: crossover and mutation. Crossover allows two solutions to mix and share parts of their solutions with each other, while mutation changes, usually with very low probability, values of a given solution before the offspring solution creation is finalized. Figure 2.3 illustrates the process of a so-called 1-point crossover, followed by mutation to generate two example offspring solutions. Selection and variation are configured so that n offspring solutions are created.

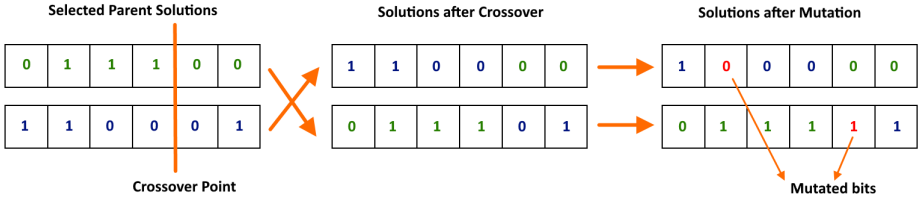


Figure 2.3: *Variation of two selected parent solutions using 1-point crossover and bit-wise random mutation. First a randomly chosen crossover point is selected, after which parts of the solutions are swapped. Then, mutation changes values of the solutions using a low probability of making a random new value assignment to each variable independently. This ultimately results in two offspring solutions.*

The sGA, though simple in its design, has proven very useful and efficient for many academic and real-world problems, and is often considered a cornerstone of evolutionary computation. A big drawback of this relatively simple approach however, is its inability to automatically adapt the variation operators to process problem structure such as variable dependencies. Because such information is not considered in the classical recombination and mutation operators associated with sGA, these operators can be very disruptive in the sense that combinations of values for strongly dependent variables are copied from parents to offspring with very low probabilities, making it difficult to find high-quality solutions if many such sets of strong

dependent variables exist. This, and other optimization challenges related to EAs are detailed in the next section.

2.1.1 Optimization Challenges

Because many core concepts are shared across the entire field of evolutionary computation, a common set of challenges can be identified, despite major differences in various EA implementations.

Exploration versus Exploitation

One of they key challenges is the management of so-called exploration versus exploitation [32] [91]. Exploration refers to the ability of EAs to examine different, previously unconsidered areas of the search space, potentially even far away from where the first solutions were initialized. The ability to explore effectively can be of great benefit if the global optimum is hard to find, e.g., due to the presence of (many) local optima, or if the global optimum is located far away from where the population is located in the search space. It can however also be a hindrance to the optimization as too much exploration can lead to needless and inefficient search in regions of the search space which are not interesting. Exploitation is effectively the opposite to exploration, as it describes the ability to focus the search in a given area and efficiently find high-quality solutions in that area. While exploitation may lead to improved ability of reaching a local optimum, it may also deter the search from other promising areas in the search space where a potentially better local optimum, or in fact the global optimum may be located. While there is no easy answer to how much exploration versus exploitation should be performed by an algorithm, the interplay between factors such as the population size p, and the mechanisms used for the selection and variation phases of execution are crucial to reaching an efficient balance. The challenge of efficiently balancing the exploration and exploitation capabilities of an algorithm becomes arguably even larger when working with mixed-integer problems. In this thesis, we will introduce mechanisms specifically aimed at improving the balance of exploration and exploitation when solving mixed discrete-continuous optimization problems and utilize them in GAMBIT.

Parameterization

Even for the sGA, many parameters require consideration before optimization can take place. Population size, number of solutions involved in the selection phase, and probability of mutation are just some of the parameters that need to be determined. If the type of problem being solved is well-understood, it may be possible to competently pre-determine these parameters. However, in case of black-box optimization, often encountered in the real-world, it may be impossible to set the parameters so that efficient optimization is achieved without further analysis of the problem, or much trial-and-error. To deal this issue, we will design GAMBIT such that the need to set any parameters prior to execution is essentially removed.

Variable Dependencies

For many non-trivial problems, certain variables will be dependent on each other. That is to say that if all variables were to be independent, they would each have a separate contribution to the fitness function, making the problem relatively easy to solve as each variable can then be set optimally simply by considering each variable separately and determining the best value just one variable at a time. Especially if the variables are binary or low-cardinality integer, the problems becomes trivial to solve. Intuitively, the existence of variable dependencies means that a change to some problem variables, without consideration of their dependencies, could have a very negative effect on the optimization effectiveness of an EA. In cases where dependencies exist between problem variables, sGA can become very inefficient [86]. This is because the crossover operator picks the crossover-point at random without consideration of underlying problem structure, leading to potentially very disruptive behavior and exponential scale-up of the population size. A proper crossover, can however be effective and efficient [43].

The proven benefits of properly considering problem structure in terms of variable dependencies, led to materialization of a novel type of EA, namely the model-based Evolutionary Algorithms. Such algorithms utilize the core concepts of EAs, but they infuse this process with the ability to learn and exploit the underlying problem structure. Especially in a black-box setting, underlying variable dependencies can have a strong influence on to how difficult a problem is to optimize. The challenges this poses, are potentially even more difficult when faced with mixed-integer problems, as variable dependencies may then span across different domains. Effectiveness in dealing with the latter is central to the design of GAMBIT.

2.2 Model-Based Evolutionary Algorithms

Model-based EAs are a specific type of EAs, which attempt to learn and represent characteristics of the optimization problem in a model that can be exploited during variation. Among these MBEAs, many of them focus on capturing and processing variable dependencies, a key type of complexity-imposing problem characteristics, in order to improve the optimization performance. Gene-pool Optimal Mixing Evolutionary Algorithms (GOMEAs) [88] are a relatively recent representative of this class of algorithms. By introducing a new means of defining and exploiting important building blocks of a problem, GOMEAs utilize the central GA notion of exchanging (good) parts of solutions approach in a potentially much more powerful way. Estimation of Distribution (EDA) [55] [68] [60] algorithms utilize statistical model building mechanisms to model and process variable dependencies. Where the former were originally introduced and proven effective for discrete solution representations, the latter has various state-of-the-art instances for continuous variables. In this section we detail an algorithm from the GOMEA family, as well an EDA, which are powerful in their respective domains, and are used to provide many components of GAMBIT.

2.2.1 Linkage Tree Genetic Algorithm

The Linkage Tree Genetic Algorithm (LTGA) is a member of the Gene-pool Optimal Mixing Evolutionary Algorithm (GOMEA) family [88] and was introduced for binary variables. For any GOMEA instance, to generate a new solution, a collection of subsets of problem variables, known as the Family of Subsets, or FOS, is traversed and each subset is used in a mixing operation with existing solutions. The rationale behind using a FOS structure is to explicitly identify and exploit subsets which make up important building blocks. Generally, FOS subsets could be composed of any number of variables. These subsets can be acquired in a variety of ways ranging from always enforcing completely independent subsets of one variable each, or otherwise predetermined subsets, to some learning procedure. In LTGA, hierarchical clustering is employed to arrive at a so-called linkage tree that expresses a hierarchical clustering of variable dependencies, ranging from independent to fully dependent. Each node of this tree is considered a FOS subset. An example of a linkage tree and the associated FOS is given in Figure 2.4.

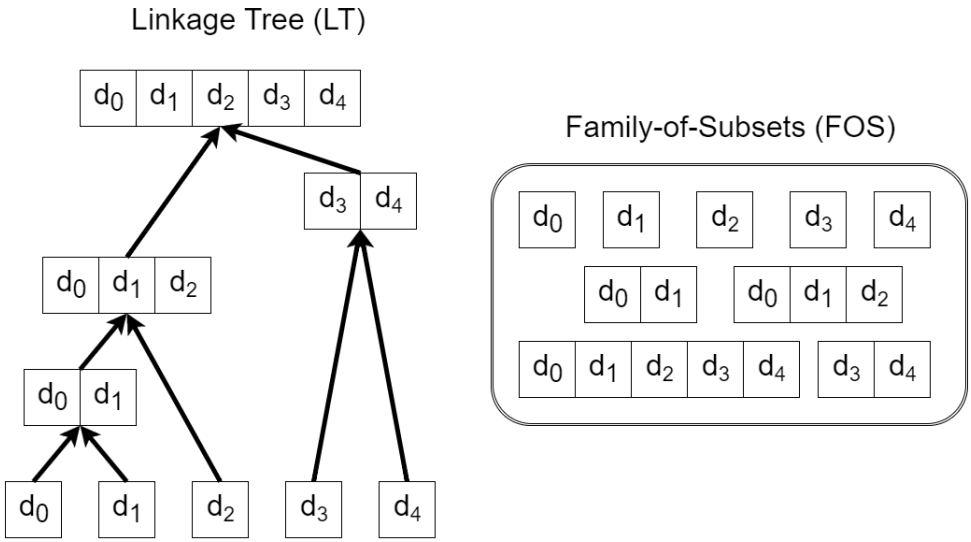


Figure 2.4: *A linkage tree created with a hierarchical clustering algorithm, and its associated FOS structure.*

To perform hierarchical clustering, mutual information is used as the basis for a distance measure. A tree is built from the bottom-up, where each node represents a subset of problem variables. This process starts from singleton clusters that are leaf nodes of the linkage tree, where each cluster contains only one problem variable. In more recent versions of LTGA, an unweighted pair group method with arithmetic mean (UPGMA) is used as a distance metric based on pairwise dependencies, not full MI [84][37]. Building a LT with this distance metric improves performance, as it can be done in $O(nl^2)$ time.

Each linkage tree node represents a potential dependency between some subset

of problem variables. As illustrated in Figure 2.4, each linkage tree node becomes a subset of the FOS. Compared to discrete EDAs, that came before LTGA, more information can now be obtained in less runtime, while still having only a linear number of subsets, as exactly $2l_d - 1$ subsets are created each time where l_d represents the number of discrete variables.

The general procedure of LTGA is as follows. First, an initial population of N randomly generated solutions is created and each solution is evaluated. In each generation, the linkage tree is built using hierarchical clustering. Next, every solution in the population is considered (in turn). For each solution, a copy is created to start the creation of an offspring solution. Each of the subsets in the FOS is then considered in a random order. For each of these subsets, a random donor solution from the current population is selected. Variable values specified by the current FOS subset of the donor solution are copied onto the corresponding variables of the current offspring solution. This mixing process alters the offspring solution, which is immediately evaluated. If this mixing, depicted in Figure 2.5, results in an equally good, or better (a contribution first made by the work in Chapter 3 of this thesis), solution than the solution before mixing, the new offspring solution is saved. Otherwise, the changes are discarded. If no improvement is found after processing all of the FOS subsets, a Forced Improvements (FI) mechanism is invoked. In FI the mixing procedure is repeated, using as the donor the best solution currently available in the population. FOS subsets are processed again and once an improvement is found for a subset, the solution is accepted and the remaining subsets are omitted. FI have been shown to improve the convergence of solutions, without overly increasing the selection pressure ([18]). After each solution in the population is processed in this fashion, the offspring solutions replace the parent population.

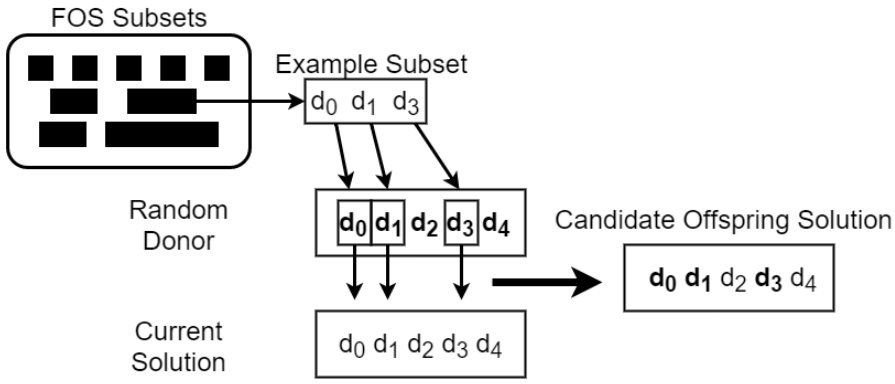


Figure 2.5: *Creating offspring solution with a FOS subset.*

LTGA achieves very good results on problems such as Deceptive Trap, NK-Landscapes, MaxCut, and Hierarchical Functions [18] as well as MAX-SAT [77] (the latter being a contribution made by the work in Chapter 3 of this thesis).

2.2.2 iAMaLGaM

The estimation of distribution algorithms (EDAs), have their roots in discrete representations, but are today perhaps best known for the state-of-the-art algorithms in this class for continuous optimization. EDAs generate selection pressure by selecting a fraction of best solutions currently present in the population. Unlike other EAs, EDAs utilize information from all selected solutions at once, through the use of a probability distribution over all of the solutions. These distributions are estimated from the selected solutions and used to sample offspring solutions [68].

The Incremental Adapted Maximum-Likelihood Gaussian Model Iterated Density Estimation Evolutionary Algorithm (iAMaLGaM) is a state-of-the-art Estimation-of-Distribution Algorithm (EDA) for real-valued black-box optimization [15]. Following the general EDA paradigm, iAMaLGaM estimates a Gaussian distribution every generation from the selected solutions and generates new solutions by sampling the estimated distribution. The mean vector and covariance matrix are estimated incrementally using intergenerational memory decay on maximum-likelihood estimates. A mechanism which scales up the values in the co-variance matrix when needed (based on the average improvement) is used to counteract the risk of premature convergence on slope-like regions of the search space. This mechanism works in tandem with the intergenerational Anticipated Mean Shift (AMS). AMS is a multi-generation spanning mechanism responsible for considering and adapting to the direction the estimated distributions shift over generations. The optimization performance of iAMaLGaM has been shown to be invariant to rotations of the search space (if the full covariance matrix is used) and to scale-up very well on many well-known black-box optimization benchmarks [16] [17].

The general procedure of iAMaLGaM is consistent with the EA paradigm and can be summarized as follows. Initially, a random population of solutions is created. If known, an initialization range can be specified for problem variables. All variables in each solution of the initial population are randomly generated from this range, and the evaluation function is used to calculate the fitness of each solution. A selection typically spanning the top 35 percent of best solutions, based on the fitness values determines a set of solutions which are used to estimate the parameters of the updated distribution, as outlined above. The updated distribution can now be used to sample an entire new population of solutions. Once all new solutions have been sampled, these new solutions replace the population. Typically, the best solution is copied from the current generation to the next. A general visualization of the process can be found in Figure 2.6.

2.3 Summary

While the field of evolutionary computation has significantly advanced over the course of past years, core concepts have remained similar ever since the introduction of the simple Genetic Algorithm. Key concepts include maintaining a population of solutions and the use of selection and variation operators in a generational process. Likewise, key challenges to the design of effective and efficient EAs remain the same though many strides have been made to overcome these challenges for dif-

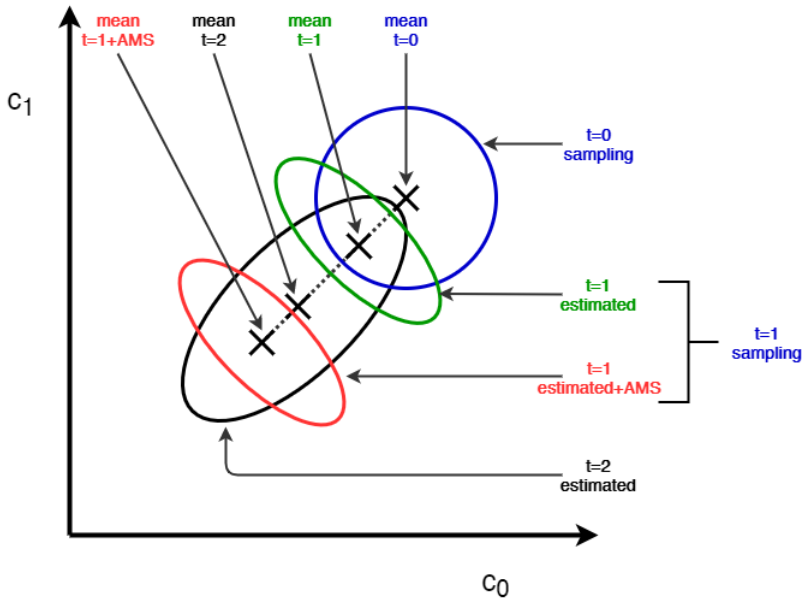


Figure 2.6: *Simplified example of iAMaLGaM execution for 2 subsequent generations for a specific 2-variable minimization problem, the 2D slope:* $c_0 + c_1$.

ferent settings (e.g., types of optimization problem). Trade-offs between exploration and exploitation, efficient ways to reduce the number of parameters to be set by a user without (drastically) losing performance, or ways to handle problems where variable-dependencies are present are some of the core challenges in evolutionary computation.

Model-based evolutionary algorithms have more recently become a prominent approach to effectively address some of such challenges. Two state-of-the-art EAs, LTGA and iAMaLGaM, for the discrete and continuous domain respectively, have been shown to provide many advantages over simpler approaches in terms of scalability and the ability to address variable dependencies, especially when faced with black-box optimization problems with exploitable underlying structure.

The observed performance of LTGA and iAMaLGaM motivates their potential applicability to the mixed-integer domain. While mixed-integer optimization introduces many new challenges, they are inherent discrete and continuous components to the mixed-integer optimization process. Because of this, proven mechanisms of model-building EAs for individual variable domains may be highly valuable to the design of an effective and efficient mixed-integer black-box EA.

3 Gene-pool Optimal Mixing EAs for Max-Sat

This chapter is based on the following publication:

K. L. Sadowski, P. A. N. Bosman, and D. Thierens. On the Usefulness of Linkage Processing for Solving MAX-SAT. In Genetic and Evolutionary Computation Conference, GECCO '13, Amsterdam, The Netherlands, July 6–10, 2013, pages 853—860, 2013.

3.1 Introduction

How to best exploit a problem's structure is an ongoing research topic both in a black-box setting where no knowledge on the problem at hand is available and in a white-box setting where it is available. Linkage models, which attempt to describe meaningful dependencies between variables, can be powerful tools for the creation of new, highly-fit solutions. Mixing may be disruptive with respect to important building blocks otherwise [87]. A group of linkage models known as the family of subsets (FOS) has previously been introduced, and used with the Gene-pool Optimal Mixing Evolutionary Algorithm (GOMEA) framework [88]. The GOMEA framework provides an efficient mechanism for mixing partial solutions and generating new candidate solutions, given a FOS linkage model.

The FOS model represents the linkage between problem variables. Learning linkage and building a FOS model can be done in various ways including learning of variable dependencies or using a predetermined structure, when it can be derived. Within the GOMEA framework, we are interested in studying whether differently acquired FOS linkage models can have a significant impact on the fitness of generated solutions. In other words, when is the overhead of exploring linkage information useful ?

In this chapter we study the Maximum Satisfiability (MAX-SAT) domain to analyze the effects of using different linkage models. The definition of the Satisfiability Problem (Section 3.2) permits us to take two separate approaches: black-box and white-box.

In the black-box setting, we assume no problem-specific information is known. We examine a GOMEA instance which dynamically learns the FOS linkage model by generating a linkage tree. This Linkage Tree Genetic Algorithm (LTGA) has already been successfully used to solve problems such as the Nearest Neighbor NK-landscape or Deceptive Trap Functions with high efficiency [88]. We study whether the benefit of learning linkage extends to solving MAX-SAT problems. We also study the effects of complementing the LTGA with a local search, and contrast the results with another black-box algorithm, hBOA [69].

From the white-box optimization perspective, more of the SAT problem's information is exploitable. In addition to learning the linkage with LTGA, we introduce a new GOMEA instance: SAT-GOMEA. This algorithm predetermines the FOS linkage structure by exploiting the structural information from the SAT problem's formula definition. We show that for SAT problems known to posses an underlying structure, using linkage information yields significantly better solutions than for GOMEA instances which ignore linkage. We also present the benefits of combining LTGA and SAT-GOMEA with the well-known Walksat LS in the white-box setting.

While the main goal of this chapter is to establish the significance of exploring variable linkage in the GOMEA instances, we also wish to study how our hybrid LTGA+Walksat and SAT-GOMEA+Walksat algorithms compare to SAT solvers Walksat and GSAT. Local search (such as GSAT, or Walksat) in the MAX-SAT domain have been reported by Rana and Whitley [72] to perform better than GAs. We wish to show how LTGA and SAT-GOMEA behave on both, uniformly distributed and structured MAX-SAT benchmarks, when combined with LS algorithms.

The remainder of this chapter is organized as follows. In Section 2, we provide background information on linkage models and GOMEA. Then we define the Maximum Satisfiability problem and the benchmarks used in our experiments. In Section 3, we describe the black-box approach. Different FOS linkage model-building algorithms, as well as the experimental results are presented and discussed. In Section 4, we examine the white-box setting. We explain the SAT-GOMEA algorithm, and analyze the effects of different linkage models. We also provide experimental results of the hybrid LTGA+Walksat and SAT-GOMEA+Walksat algorithms. Section 5 summarizes and concludes our findings.

3.2 Background

The Gene-pool Optimal Mixing Algorithm (GOMEA) is a framework which we use to explore the benefits of considering linkage information in the MAX-SAT domain. This section provides background on GOMEA and how the family of subsets (FOS) linkage model is used to create GOMEA instances. We also explain the MAX-SAT problem and present the SAT benchmarks used in our experiments.

3.2.1 Linkage Models and GOMEA

A linkage model has the capacity to represent groups of variables for a given problem, which make an important contribution to the fitness of solutions. In this chapter,

we use the family of subsets (FOS) linkage model to represent variable dependencies. A FOS is a set of subsets of a main set, which contains all problem variables [89]. Essentially each subset of the FOS contains between one and $l-1$ items, where l is the number of problem variables. The number of subsets, their length and composition is determined based on the different FOS model-building algorithms. The difference between GOMEA instances used in this chapter lie in how the FOS linkage model was determined.

GOMEA is a family of genetic algorithms which generate new solutions through mixing, given a FOS structure [89]. Initially, a random population of solutions is created and evaluated. Then, for each solution, the mixing process takes place. Each subset of the FOS model is used as the variation operator, or a mask, which is applied to copy values from a randomly picked donor solution from the population to the cloned solution and the effect is evaluated immediately. If for a given subset, the mask applied from the donor to the solution generates a higher-fitness solution, this new solution is kept. If it does not result in a better solution, the mask is rejected. For every solution, this mixing continues until all the FOS model masks have been tested. It guarantees that the generated candidate solutions are not worse than the original parent. If the solution is not improved after all the FOS masks are applied, Forced Improvement (FI) takes place [18]. The mixing process is repeated for this solution, but this time the donor is the best available solution in the population, instead of a random one. If this process also fails at improving the solution, this solution is entirely replaced with the best solution from the population. Ultimately, the set of newly generated solutions replaces the parent population entirely. Pseudo-code of this algorithm can be found in Figure 3.1.

3.2.2 Problem Definition

The Boolean Satisfiability (SAT) Problem is a well-known NP-complete problem. Its instance consists of a set of Boolean variables $X = \{x_1, x_2, x_3, .., x_l\}$, as well as a Boolean formula $F : \{0,1\}^n \rightarrow \{0,1\}$. This formula is a logical conjunction of clauses of the form $c_1 \wedge c_2 \wedge c_3 \wedge ... \wedge c_m$, where each clause c_i is a logical disjunction of a subset of the problem variables (or its negations).

An example clause would be $c_i = (x_1 \vee x_2 \vee \neg x_4)$ etc. An example of a complete formula with l=3 variables and m=3 clauses could look like $(\neg x_2) \wedge (x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_2)$.

The objective in solving a SAT Problem is to find a variable assignment $v : X \rightarrow \{0,1\}$, which will result in the Boolean formula evaluating to True. MAX-SAT is a generalization of the SAT problem. The objective of a MAX-SAT problem is to maximize the number of clauses which evaluate to True. In this chapter the number of unsatisfied clauses (false clauses) is a measure of fitness, where zero refers to a fully satisfied formula. This means that in our experiments our goal is to minimize the fitness value.

GOMEA *//population size n, problem size l*
for $i \in \{0, 1, \ldots, n-1\}$ **do**
  $\mathcal{P}_i \leftarrow$ CREATERANDOMSOLUTION()
  EVALUATEFITNESS($\mathcal{P}_i$)
while ¬TERMINATIONCRITERIONSATISFIED **do**
  $\boldsymbol{x}^{best} \leftarrow arg\ \max_{\boldsymbol{x} \in \mathcal{P}} \{fitness[\boldsymbol{x}]\}$
  $\mathcal{S} \leftarrow$ TOURNAMENTSELECTION($\mathcal{P}, n, 2$)
  LEARNMODEL($\mathcal{S}$)
  for $i \in \{0, 1, \ldots, n-1\}$ **do**
    $\mathcal{O}_i \leftarrow$ FI-GOM($\mathcal{P}_i$)
  $\mathcal{P} \leftarrow \mathcal{O}$

FI-GOM($\boldsymbol{x}$)
$\boldsymbol{b} \leftarrow \boldsymbol{o} \leftarrow \boldsymbol{x}$; $fitness[\boldsymbol{b}] \leftarrow fitness[\boldsymbol{o}] \leftarrow fitness[\boldsymbol{x}]$; $improved \leftarrow false$
for $i \in \{0, 1, \ldots, |\mathcal{F}|-1\}$ **do**
  $\boldsymbol{p} \leftarrow$ RANDOM($\{\mathcal{P}_0, \mathcal{P}_1, \ldots, \mathcal{P}_{n-1}\}$)
  $\boldsymbol{o}_{\boldsymbol{F}^i} \leftarrow \boldsymbol{p}_{\boldsymbol{F}^i}$
  if $\boldsymbol{o}_{\boldsymbol{F}^i} \neq \boldsymbol{b}_{\boldsymbol{F}^i}$ **then**
    EVALUATEFITNESS($\boldsymbol{o}$)
    if $fitness[\boldsymbol{o}] > fitness[\boldsymbol{b}]$ **then**
      $\boldsymbol{b}_{\boldsymbol{F}^i} \leftarrow \boldsymbol{o}_{\boldsymbol{F}^i}$; $fitness[\boldsymbol{b}] \leftarrow fitness[\boldsymbol{o}]$; $improved \leftarrow true$
    else
      $\boldsymbol{o}_{\boldsymbol{F}^i} \leftarrow \boldsymbol{b}_{\boldsymbol{F}^i}$; $fitness[\boldsymbol{o}] \leftarrow fitness[\boldsymbol{b}]$
if ¬$improved$ **then**
  for $i \in \{0, 1, \ldots, |\mathcal{F}|-1\}$ **do**
    $\boldsymbol{o}_{\boldsymbol{F}^i} \leftarrow \boldsymbol{x}^{best}_{\boldsymbol{F}^i}$
    if $\boldsymbol{o}_{\boldsymbol{F}^i} \neq \boldsymbol{b}_{\boldsymbol{F}^i}$ **then**
      EVALUATEFITNESS($\boldsymbol{o}$)
      if $fitness[\boldsymbol{o}] > fitness[\boldsymbol{b}]$ **then**
        $\boldsymbol{b}_{\boldsymbol{F}^i} \leftarrow \boldsymbol{o}_{\boldsymbol{F}^i}$; $fitness[\boldsymbol{b}] \leftarrow fitness[\boldsymbol{o}]$; $improved \leftarrow true$
      else
        $\boldsymbol{o}_{\boldsymbol{F}^i} \leftarrow \boldsymbol{b}_{\boldsymbol{F}^i}$; $fitness[\boldsymbol{o}] \leftarrow fitness[\boldsymbol{b}]$
    if $improved$ **then breakfor**
if ¬$improved$ **then**
  $\boldsymbol{o} \leftarrow \boldsymbol{x}^{best}$; $fitness[\boldsymbol{o}] \leftarrow fitness[\boldsymbol{x}^{best}]$

Figure 3.1: *Pseudo-code for GOMEA where FI-GOM is the genepool optimal mixing operator with forced improvement for a single solution.*

SAT Benchmarks

To show whether linkage information can improve the fitness of generated solutions we need to examine benchmarks with both, little and lots of underlying structure. Benchmarks generated from a uniformly randomized distribution are expected to have very little exploitable structure in terms of variable dependencies. Other benchmarks, such as non-uniform random, handmade or industrial benchmarks are expected to have more structure.

Empirical data is gathered over a set of benchmarks available from the SAT Problem Library: SAT-Lib [47]. Differently sized benchmarks, in terms of number of variables and clauses are considered, as well as different benchmark types: random,

handmade and industrial. Summary of all benchmarks can be found in Table 3.1. The 'unif' benchmark instance is the only instance generated randomly from a uniform distribution.

Benchmark	Abbr.	Vars.	Clauses	Type
uuf250-01	unif	250	1065	Random
sw100-X	sw100-X	500	3100	Random
am-5-5.shuffled-as.sat03-361	am-5-5	1076	3677	Industrial
hardnm-L32-03-S1527311839.shuffled-as.sat03-942	hard32	1024	4096	Random
C880mul.miter.shuffled-as.sat03-348	C880	1612	9373	Industrial
20000009987nc.shuffled-as.sat03-1665	2nc	2756	10886	Handmade

Table 3.1: *Specifications of the MAX-SAT benchmarks used in this chapter.*

3.3 Black-Box Optimization

From the black-box optimization (BBO) perspective, problem-specific information is unknown. It is a generic perspective, where only the number of variables and the evaluation function are given. In this setting, the number of evaluations performed by any algorithm in order to reach better fitness values is a frequently used measure of performance. It should be noted however, that building a linkage model also requires computational effort. This effort may be non-negligible. However, it should also be noted that the LTGA requires a relatively small computational overhead of $O(nl^2)$, compared to most higher-order model-building evolutionary algorithms such as hBOA, which requires $O(nl^2 + l^3)$ [70]. To study the behavior of different linkage models, we consider the linkage learning algorithm LTGA and contrast it against other GOMEA instances. We also compare our black-box results with the well-known hBOA EDA [69].

3.3.1 Algorithms

In this section, we examine how much influence learning linkage information has on the quality of generated solutions. To do this, we study the performance of GOMEA instances which determine the FOS linkage structure differently.

Before providing an overview of the BBO algorithms used, a small algorithmic change to the GOMEA's solution acceptance criteria is worth noting. Previous

GOMEA instances often share the same solution acceptance criteria. A newly generated solution is only accepted and introduced to the population if this new candidate solution is strictly better than its parent. We replaced this condition with a less strict one. More specifically, a solution will replace its parent if it is at least as good. In terms of a SAT problem, this allows for more plateau exploration and diversification of the population, as equally good, but different solutions are not rejected. This change has been applied to all GOMEA instances discussed in this chapter. Figure 3.2 exemplifies typical improvements achieved with this modification, which can be substantial.

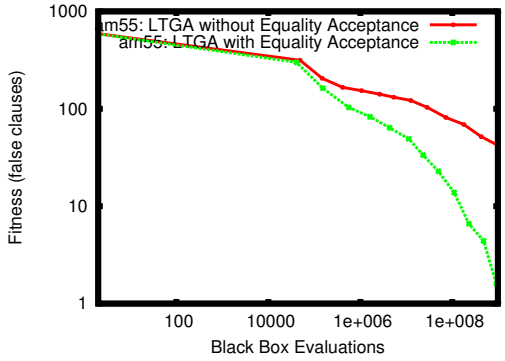


Figure 3.2: *Typical improvement from accepting equally fit solutions during mixing.*

LTGA

The Linkage Tree Genetic Algorithm is a GOMEA instance that learns the linkage between the problem variables, by generating a linkage tree at each generation. Building this tree is accomplished by a hierarchical clustering algorithm [87]. To measure dependencies between variables the hierarchical clustering algorithm uses mutual information. Based on this measure the linkage tree is build from the bottom up. Each variable is first assigned to a single cluster. Based on the mutual information new clusters are created by joining together already existing clusters. This process continues until only one cluster (containing all variables) is left, which requires an computational overhead of $O(nl^2)$ [37]. Each cluster of the linkage tree, except for the full cluster (which contains all variables), represents a subset of a FOS linkage model which is then used with GOMEA.

Random-Tree GOMEA

This GOMEA instance has a linkage structure identical to LTGA. While the FOS linkage structure is still based on a Linkage Tree, the actual linkage information is not based on learned variable dependencies. The hierarchical clustering algorithm used by LTGA is still present, and the linkage tree is still generated. However, the mutual information used to determine variable dependencies and cluster proximities is

completely random. In other words, this GOMEA instance generates it's FOS structure similarly to LTGA in terms of mechanics, however no real linkage information is discovered.

Univariate GOMEA

Univariate GOMEA is a very simple instance of the GOMEA family. All the problem variables are considered to be independent of all other variables resulting in no linkage between the variables being modeled. In other words, the FOS linkage structure only consists of singleton subsets.

3.3.2 BBO Linkage Model Experiments

Figure 3.3 shows how different algorithms from the GOMEA family perform on our representative benchmarks. For every problem and every population size considered, we perform 30 independent runs. The only algorithm in the black-box setting which attempts to learn linkage information is LTGA.

The results show that learning linkage in a uniformly distributed random problem is not very efficient with small population sizes. Initially, Univariate-GOMEA finds better solutions with less evaluations. However, with large enough population, LTGA outperforms the other algorithms.

The results for problems with a better defined underlying structure favor linkage learning. In benchmarks which are not uniformly distributed the use of linkage information shows significant improvement, even for smaller population sizes. LTGA outperforms the other algorithms. This indicates that learning variable dependencies and including those dependencies in the linkage model can be a powerful tool in the MAX-SAT domain. These results only show the advantages in terms of evaluations needed to reach near-optimal solutions. We also consider the cost of the linkage learning overhead in terms of runtime.

LTGA is the most expensive GOMEA instance, because it requires the linkage learning overhead. Table 3.2 shows the fraction of the total runtime needed by learning to reach highly-fit solutions. The results show that the time needed for linkage learning is about ten percent of the total time needed by the algorithm. The other algorithms however, may never reach good solutions, or require at least multiple orders of magnitude more time to reach similar fitness levels on the tested benchmarks. This means that the overhead of learning is acceptable in terms of runtime limitations as well.

3.3.3 hBOA and LTGA Experiments

The hierarchical Bayesian optimization algorithm (hBOA) is a well-known EDA proposed by Pelikan and Goldberg [69] as a black-box approach to solving hierarchical and nearly decomposable problems. hBOA works with a population of solutions, which are initially random. Unlike GOMEA, each generation hBOA builds a Bayesian network as a probabilistic model of promising solutions. New solutions are generated though sampling the learned network [69]. While it is a competent

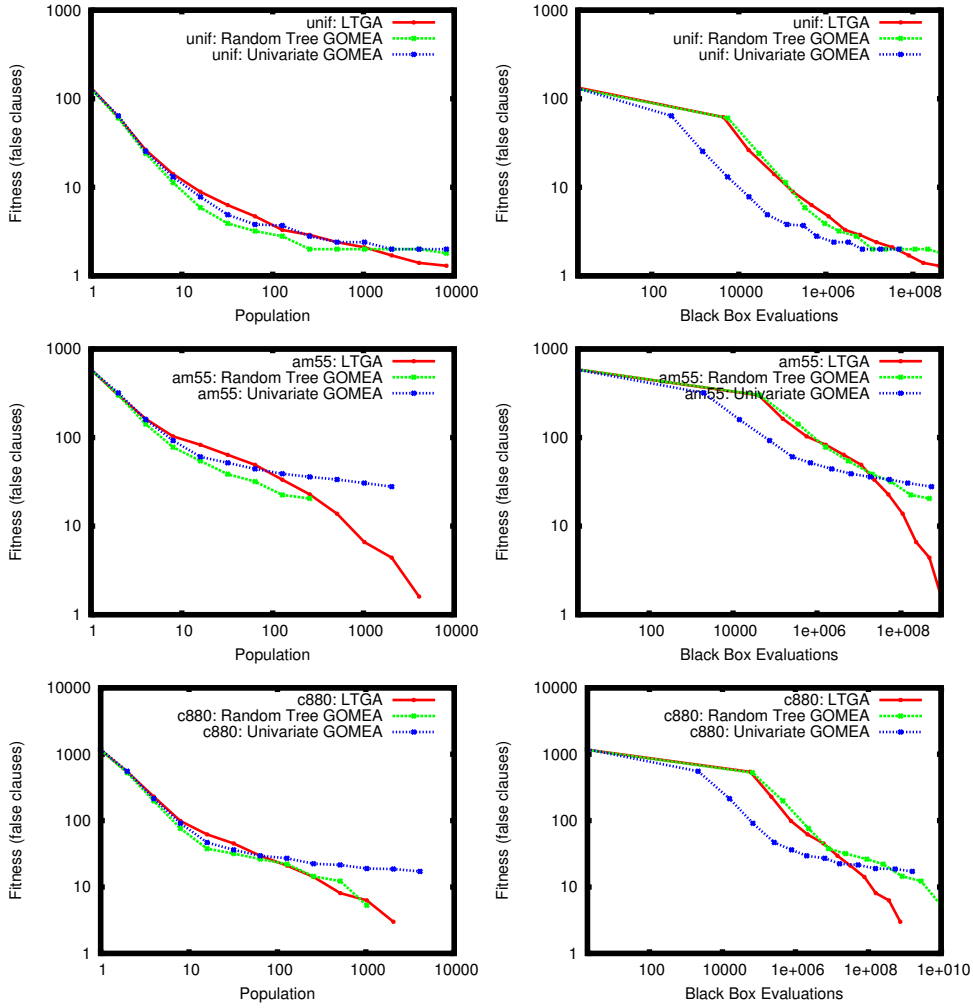


Figure 3.3: *Comparison of performances of different GOMEA instances with different FOS Linkage Models on differently structured benchmarks.*

algorithm in terms of linkage learning, the overhead of building a Bayesian network can be substantial.

hBOA was also tested on MAX-SAT. In their experiments, two local search algorithms are studied: GSAT and Walksat. GSAT is a best-improvement local search algorithm designed for the Satisfiability Problem [83]. Walksat is an extension of GSAT which incorporates random walks. Both of these algorithms are explained in more detail in Section 4. Pelikan and Goldberg show that hBOA combined with GSAT local search (hBOA+GSAT) outperforms the stand-alone GSAT and Walksat on the morphed graph-coloring problems, which are translated into MAX-SAT. These graph-coloring problems are known to be difficult for LS algorithms [69].

Instance	BBO f	St.Dev f	WBO f	St.Dev f
unif	0.093	0.012	0.116	0.019
sw100-1	0.124	0.004	0.405	0.025
am5-5	0.097	0.007	0.342	0.020
hard32	0.061	0.011	0.301	0.033
c880	0.068	0.029	0.261	0.057
2nc	0.098	0.099	0.367	0.034

Table 3.2: *Fraction f of total time needed by for linkage learning of the LTGA in BBO and WBO setting*

While GSAT and Walksat do not succeed at satisfying the tested problems within the tested runtime constrains, the hBOA+GSAT does.

In order to compare the effectiveness of LTGA, we examine a LTGA+GSAT hybrid on the same benchmark set as in [69]. It is important to note some differences between using GSAT with LTGA and with hBOA. With LTGA, a single GSAT instance is executed after mixing is complete for a given solution, which consists of many evaluations. Additionally, GSAT is allowed to run until no fitness-improving move is possible in l flips, where l is the number of problem variables. hBOA performs a GSAT search after every evaluation. It is unclear however, how many non-improving side-steps, if any, this GSAT is permitted to take.

Just like the hBOA+GSAT hybrid, LTGA+GSAT succeeded at solving the graph coloring benchmarks. Table 3.3 summarizes the results in terms of total evaluations needed. Each LTGA+GSAT result found in Table 3.3 represents the number of evaluations performed by LTGA, averaged over thirty independent runs on the smallest population size that is successful. The result show the number of evaluations of hBOA and LTGA respectively, not representing the flips performed by the local search.

3.4 White-Box Optimization

Algorithms examined in the previous section attempted optimization without any problem-specific information. However, the Satisfiability domain provides us with more information. Examining the Boolean formula structure of a SAT Problem, as defined in Section 2, allows for many adjustments which can improve the quality and the speed of generating solutions. With careful bookkeeping, partial-solution evaluations and constant-time neighborhood search steps are possible, and can be used to improve the optimization algorithm [83]. When a variable is flipped, we only need to evaluate clauses which are affected by the flip, instead of performing a full evaluation.

Since full function evaluations are no longer needed in this setting, the performance measuring criteria is changed. We measure the performance of algorithms in the white-box setting by the number of bit flips performed. This means that a single move performed by a local search is equivalent to one flip, while a mixing operation

Instance	hBOA+GSAT	LTGA+GSAT
SW100-8-5/sw100-1	1,262,018	257,823
SW100-8-5/sw100-2	1,099,761	209,972
SW100-8-5/sw100-3	1,123,012	232,916
SW100-8-6/sw100-1	1,183,518	622,748
SW100-8-6/sw100-2	1,324,857	1,388,747
SW100-8-6/sw100-3	1,629,295	1,948,879
SW100-8-7/sw100-1	1,732,697	1,433,359
SW100-8-7/sw100-2	1,558,891	1,354,856
SW100-8-7/sw100-6	1,966,648	2,659,001
SW100-8-7/sw100-7	1,222,615	917,577
SW100-8-8/sw100-1	1,219,675	1,110,319
SW100-8-8/sw100-2	1,537,094	766,796
SW100-8-8/sw100-6	1,650,568	968,124
SW100-8-8/sw100-7	1,287,180	1,322,188

Table 3.3: *Number of hBOA and LTGA evaluations in a Black-Box setting*

on a mask of length l is considered to be l bit flips. Recall, in the black-box setting the number of bit flips in a mixing operator did not matter, as a full evaluation was needed every time.

In this section, we revisit the experiments performed in the black-box setting. Will learning of linkage remain beneficial in this setting ? We also introduce a new GOMEA instance, SAT-GOMEA, whose linkage structure is predetermined, and based on some problem-specific information.

The second aspect of our white-box experimentation will take advantage of relatively cheap local search algorithms, which can be used in the Satisfiability domain. Recall that in this setting making a best-improvement bit flip can be accomplished in constant time. Because of this, many successful local searchers have been developed for the SAT-Problem. We examine the GSAT and Walksat local search algorithms and compare them with LTGA+Walksat and SAT-GOMEA+Walksat hybrids.

3.4.1 Algorithms

In addition to the equal solution acceptance modification inherited from the black-box setting, another modification to all the GOMEA family algorithms tested in this section is introduced: path remembering. A crucial feature of all GOMEA instances is generating a Family of Subsets (FOS) linkage model. Those subsets specify a subdivision of all variables which is used in the mixing process. When applying a subset mask, every bit in the mask needs to be changed according to the material in the donor solution [89]. The path remembering modification allows for looking at every intermediate solution which was generated while applying a given subset mask. If one of the intermediate solutions is more fit than the one generated from applying the full mask, this intermediate solution is kept instead. Using path remembering is possible in the white-box setting, because the bit flip cost of evaluating a full mask

is identical to traversing every bit flip of this mask individually in random order. This is not the case in the black-box setting. In the a black-box setting, using a full mask costs one full function evaluation, however performing each bit flip individually would require a full evaluation at every step. Figure 3.4 exemplifies the typical improvements achieved by introducing path remembering. Since the difference in terms of bit-flips used may not be clear on a log-scale graph, a t-test is used to confirm statistical significance with p-value less then 0.001.

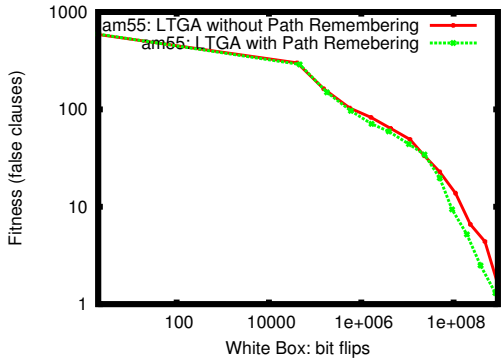


Figure 3.4: *Typical improvement from allowing Path Remembering during solutions mixing in the white-box setting.*

We study the linkage model effects from the white-box perspective using four different GOMEA instances. The first three have already been introduced in the black-box setting: LTGA, Random-Tree GOMEA and Univariate GOMEA. In this section we introduce another GOMEA instance: SAT-GOMEA, which generates a predetermined linkage model. We also explain two local search algorithms for solving SAT problems: GSAT and Walksat. Finally, we present a hybrid configuration of LTGA+Walksat and SAT-GOMEA+Walksat.

SAT-GOMEA

SAT-GOMEA introduces a predetermined linkage model. It is a strictly white-box algorithm, as it requires problem-specific information which is not available in the black-box setting. SAT-GOMEA uses a predetermined FOS linkage model which is directly related to the SAT-Problem's Boolean formula. Specifically, the FOS linkage model of this algorithm consists of subsets mapped directly from the clauses of the Boolean formula. Each clause of the formula is represented by a subset in the linkage model and each variable in a clause is represented by the same variable within a subset. For example a clause $c_i = (x_1 \vee x_2 \vee \neg x_4)$ would be represented by a subset of the form $s_i = \{x_1, x_2, x_4\}$

Local Search Algorithms

GSAT and Walksat algorithms are well-known Satisfiability problem solvers. They are very simple and fast, yet surprisingly well-performing algorithms. GSAT is a best-improvement local search algorithm. If there is no actual improvement possible, a sideways step is taken by choosing a random bit flip which does not change the fitness value [83]. Walksat consists of a GSAT move and a walk move which are performed with probability p and $1-p$ respectively. GSAT moves consist of performing a best possible (fitness-improving) single bit flip. While a GSAT move is not fully deterministic because tie breaks between different solutions with the same best fitness improvements and sideways steps are resolved randomly, it is the walk move that really brings a stochastic factor to the Walksat algorithm. During a walk move, a variable from a randomly chosen unsatisfied clause is flipped [82]. While the GSAT move attempts to always improve the current solution, the walk move allows for a greater search-space exploration. Throughout our experiments, we use the default p value of 0.5 for Walksat.

Hybrids

LTGA+Walksat and SAT-GOMEA+Walksat are two hybrid algorithms which combine modeling linkage information, genetic solution mixing and relatively inexpensive and efficient local search. After a solution is generated by the respective genetic algorithm, it is subject to a short local optimization performed by Walksat. We wish to show that on many benchmarks which contain underlying structures, using a hybrid algorithm can generate results competitive with stand-alone GSAT or Walksat in terms of best fitness found and the number of flips performed, within our experimentation limits.

3.4.2 WBO Linkage Model Experiments

We wish to examine if linkage information is still a factor from the white-box optimization perspective. Recall, in the black-box setting the Linkage Tree Genetic Algorithm performed significantly better than algorithms which did not use linkage information in their FOS structures. This time we have the LTGA and SAT-GOMEA algorithms which use some linkage information, and Random-Tree GOMEA and Univariate GOMEA which ignore the linkage. Results in Figure 3.5 correspond with the black-box results. On a uniformly generated random benchmark, exploring the linkage by learning, or using a predetermined structure is an unnecessary overhead with small population sizes. With large enough population budget however, benefits of linkage information become clear. On more challenging and structured benchmarks this advantage becomes clear sooner. Algorithms using linkage information find near-optimal solutions faster in terms of bit-flips.

We again also considered runtime. We measured how much the learning overhead adds to the runtime of the algorithms. LTGA is the most expensive GOMEA instance, because it requires the linkage learning overhead. Table 3.2 shows the fraction of the total runtime needed by learning to reach highly-fit solutions. The results show that learning becomes relatively more expensive in the white-box setting. This

was anticipated, as the rest of the algorithm can perform faster in this setting. However, the time needed for learning remains a constant factor, which did not exceed more than 45 percent of the total runtime on any of the tested benchmarks. While it is possible to construct a benchmark where this conclusion would not hold (very many variables, very little clauses) , for most interesting benchmarks (near the phase-transition) the learning overhead should remain relatively small due to the clause-to-variable ratio. The results show that on those benchmarks GOMEA instances which do not use linkage information may never reach, or require multiple orders of magnitude more time in order to reach the same levels of fitness, again favoring the use of building linkage models, even in the white-box setting.

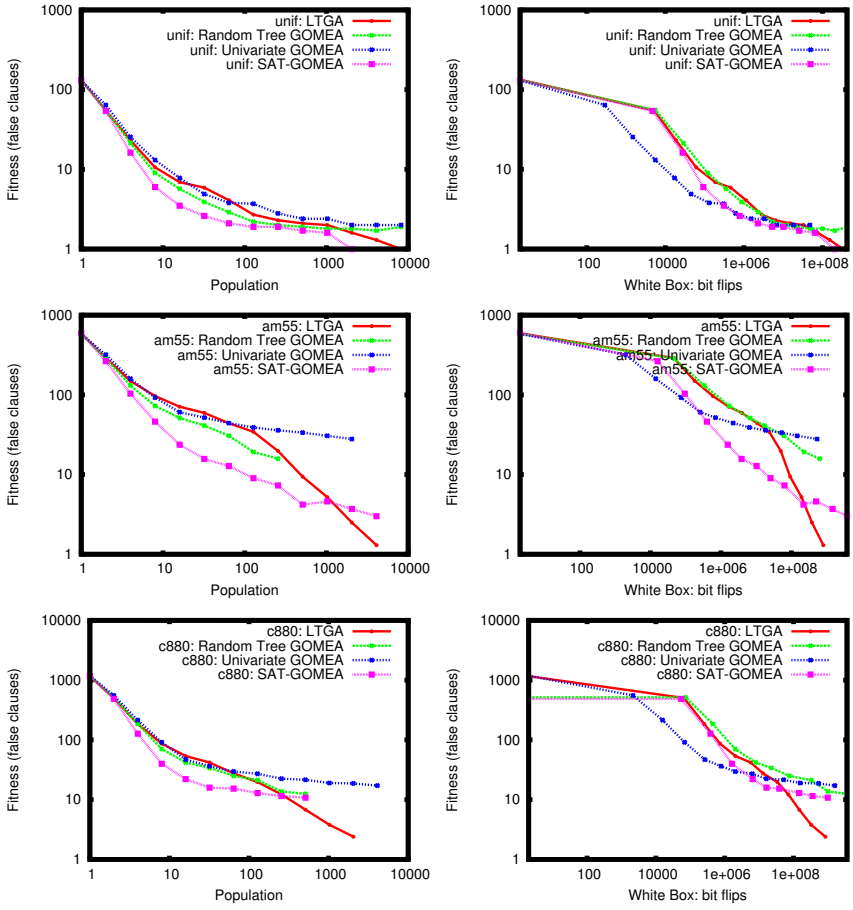


Figure 3.5: *Comparison of performances of GOMEA instances on differently structured benchmarks in the White Box Optimization setting.*

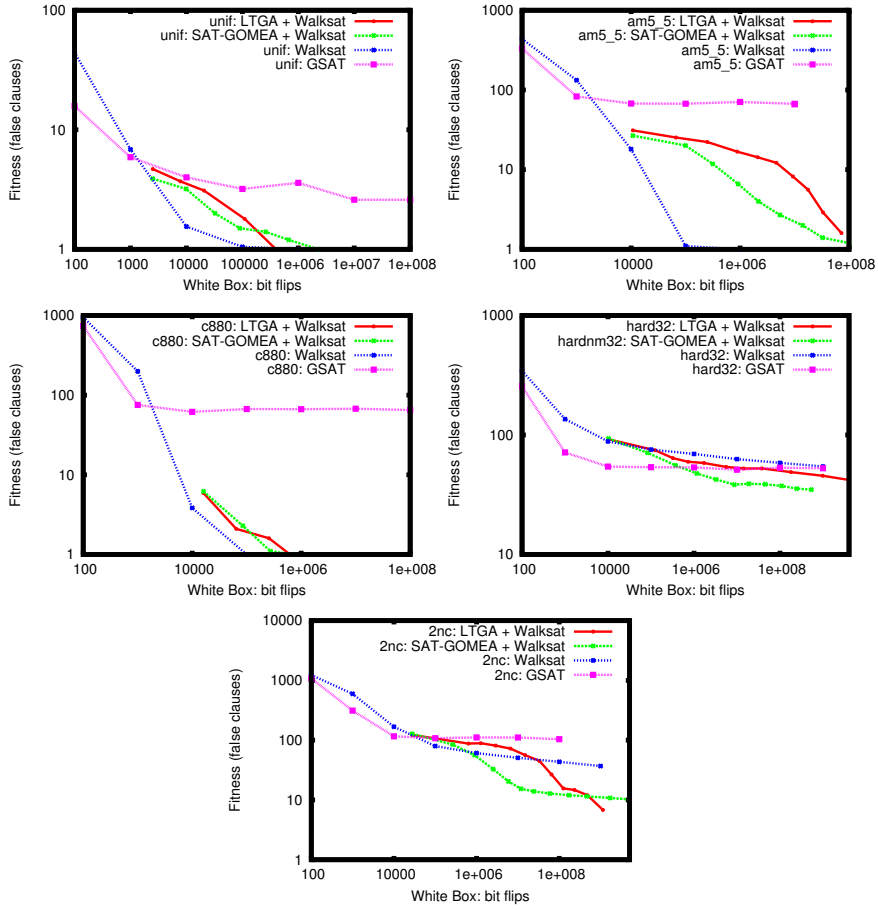


Figure 3.6: *Comparison of performances of the LTGA+Walksat, GOMEA-SAT+Walksat and stand-alone Walksat and GSAT algorithms*

3.4.3 Hybrid Experiments

So far in the white-box setting we only described experiments performed with the GOMEA instances exclusively. While the results are promising, it does not help us understand how these algorithms perform in comparison to other SAT solvers. Figure 3.6 shows the LTGA and SAT-GOMEA algorithms enhanced with a Walksat LS. After any new solution was generated by one of those GAs, Walksat was allowed to optimize it further for $5 * l$ flips, where l is the number of variables in a problem. The $5 * l$ search length was chosen, as it generated good results and balanced the number of flips performed by the GA and local searcher well.

It is not a surprise that on many simple and uniformly structured problems the stand-alone Walksat performs better. In fact, even on more structured but relatively simple benchmarks it arrives at the optimal solution faster. However, as the stand-

alone GSAT or Walksat algorithms fail to reach good solutions on more complex benchmarks, the LTGA and SAT-GOMEA hybrid versions continue to find better solutions with the same number of bit-flips.

It remains very benchmark-specific which, LTGA or SAT-GOMEA, performs better in terms of required bit-flips. However, the results show that using linkage information in the GOMEA setting successfully helps to guide Walksat into promising search space locations. By doing so, Walkast can discover solutions which it failed to encounter on its own. Moreover, given that no additional computation overhead is required for building the linkage model in SAT-GOMEA, it is preferable or at least worth trying.

The Average Landscape-Guided Hopping (ALGH) algorithm was recently introduced for MAX-SAT [71]. ALGH uses a population of independent GSATs, whose results are averaged and generate a single solution. This solution is then used as starting point for Walksat. ALGH outperforms GSAT and Walksat, however reported experiments are limited to uniformly distributed MAX3SAT instances above the phase transition. Our experiments show that obtaining linkage information can be very useful on problems which are not uniform, and possess underlying structure.

3.5 Conclusions

We have discussed the behavior of instances from the Gene-pool Optimal Mixing Algorithm family in the MAX-SAT domain. The algorithms differ in the fashion in which the FOS Linkage Model was created. In the generic black-box optimization setting, LTGA, which obtains its linkage model by learning and constructing a linkage tree, was proven superior over other GOMEA instances which did not examine linkage on benchmarks of sufficient complexity.

We were unable to determine if the linkage structure predetermined with SAT-GOMEA performed better than one learned by LTGA. The results varied based on the benchmark tested. However, exploitation of linkage information proved to be just as beneficial in the white-box setting. Algorithms which build linkage models based on some structural knowledge, the linkage learning LTGA and the predetermined linkage SAT-GOMEA, performed better than GOMEA instances which lacked linkage information. This lets us conclude that for benchmarks with some underlying structure and sufficient difficulty, the exploration of the linkage generates better-fit solutions faster. The results hold in terms of evaluations performed in both black and white-box settings, and in terms of the total runtime the algorithms need to generate near-optimal solutions.

To compare those GAs with other existing solvers, we created a LTGA+Walksat and SAT-GOMEA+Walksat hybrids. LS algorithms (such as GSAT, and Walksat) in the MAX-SAT domain have been reported to perform better than GAs. Our results confirm that LTGA does not outperform algorithms such as Walksat on uniformly randomized MAX-SAT problems and other benchmarks which are easily solvable by local searchers. However, in cases of more structured and difficult benchmarks which cause local searchers to stall, both of our hybrid algorithms continued to improve their solutions.

Extensive research is ongoing in the SAT-problem domain. It is more than likely that for problems where the LTGA or SAT-GOMEA do not perform well, algorithms already exist which can solve these problems more efficiently. The optimal structure of a SAT problem for use within GOMEA is still not known, and research on further improvements is possible. We did however succeed in showing that exploring the structure of some satisfiability problems is beneficial. Furthermore, our results added evidence to the idea that the use of linkage information could be a useful optimization technique for a broader range of problems, both in the black-box and white-box setting.

4 Mixed-Integer Optimization and Benchmarks

This chapter is based on the following publication:

K. L. Sadowski, D. Thierens, and P. A. N. Bosman. Combining Model-based EAs for Mixed-Integer Problems. In Parallel Problem Solving from Nature – PPSN XIII, pages 342—351, 2014.

The presence of both discrete and continuous variables introduces new optimization challenges. Scalability in the face of non-trivial dependencies and unbalanced numbers of variables with different domain sizes are some of the key challenges unique to the mixed-integer (MI) domain. In order to properly experimentally analyze any mixed-integer approach, a selection of benchmark problems is necessary that covers the type of difficulty that we wish to be able to tackle efficiently and effectively. In this chapter we therefore introduce and consider a set of benchmarks problems, which take into account factors that play a key role in affecting the complexity of mixed-integer problems. We identify these factors to include the problem size, problem composition in terms of the proportionality of discrete and continuous variables, and the different types of variable dependencies that can be present in mixed-integer problems, i.e., combinations of discrete-only, continuous-only and mixed discrete-continuous.

In order to consider these factors, we consider a set of well-known discrete and continuous functions that exhibit different types of landscape features in-line with the key factors mentioned above. In Section 4.2 we introduce a set of mixed-integer benchmarks that combine these discrete and continuous functions in various ways, resulting in problems with different types of intra- and inter-variable dependencies and problem size and composition, which will be used throughout this thesis.

4.1 Mixed-Integer Optimization

Recall that a mixed-integer problem can be defined as follows:

$$min \;\; f(\boldsymbol{x}) = f(\boldsymbol{x_d}, \boldsymbol{x_c}) \tag{4.1}$$

$$s.t. \;\; h(\boldsymbol{x}) = \mathbf{h}(\boldsymbol{x_d}, \boldsymbol{x_c}) = 0, \;\; g(\boldsymbol{x}) = \mathbf{g}(\boldsymbol{x_d}, \boldsymbol{x_c}) \leq 0 \tag{4.2}$$

where $\boldsymbol{x}$ represents the solution

$$\boldsymbol{x} = \boldsymbol{x_d}\boldsymbol{x_c} = d_0 ... d_{l_d - 1} \; c_0 ... c_{l_c - 1} \tag{4.3}$$

where $d_i \in D \in \mathbb{N}, c_i \in C \in \mathbb{R}$ and $\boldsymbol{x_d}$, $\boldsymbol{x_c}$ are the groups of all discrete and continuous variables, respectively. In this thesis, although it does not affect generality, $D \in \{0, 1\}$. f is the objective function. **h** and **g** are the sets of equality and inequality constraint functions respectively. If both of these sets are empty, the Mixed-Integer problem is considered to be unconstrained.

4.2 Mixed-Integer Problems

We introduce a set of functions primarily designed to test the dependency-structure learning abilities of the algorithms under study. The rationale behind creating these problems is to introduce mixed-integer problems with a well-understood, customizable structure, with different types of intra- and inter-variable dependencies. These types of problems will allow us to get more in-depth insight into the performance of model-based EAs when faced with different types of problem landscapes with respect to variable dependencies, problem sizes and problem composition.

The next Section explains and summarizes the discrete and continuous functions which we use as components of the mixed-integer problems. Following this overview, we provide an explanation of the design of the mixed-integer functions which include different types of intra- and inter-variable dependencies and will be used as benchmark problems throughout this thesis.

4.2.1 Problem Components

To design the mixed-integer benchmarks we use some well-established benchmark problems and adapt them into the mixed-integer setting by using them as components that comprise a mixed-integer problem. In all problems, minimization is assumed. Definitions of the well-established benchmark functions can be found in Table 4.1. Note that due to minimization and by our design, zero is the optimal value for all proposed functions. Detailed explanation of each of these functions is provided below.

Sphere Function

In the Sphere function all problem variables are completely independent from each other. It is defined as follows:

$$F_{Sphere}(\boldsymbol{x_c}) = \sum\nolimits_{i=0}^{l_c - 1} c_i^2 \tag{4.4}$$

Table 4.1: *Continuous and Discrete functions that are used to define our MI benchmarks*

Function Name	Domain	Definition
Sphere	Continuous	$F_{Sphere}(\boldsymbol{x_c}) = \sum_{i=0}^{l_c-1} c_i^2$
Rotated Ellipsoid	Continuous	$F_{R.Ellip.}(\boldsymbol{x_c}) = F_{Ellip.}(\mathbf{R} * \boldsymbol{x_c})$, where $F_{Ellip.}(\boldsymbol{x_c}) = \sum_{i=0}^{l_c-1} 10^{6*i/(l_c-1)} * c_i^2$
Onemax	Discrete	$F_{Onemax}(\boldsymbol{x_d}) = \sum_{i=0}^{l_d-1} 1 - d_i$
Deceptive Trap	Discrete	$F_{DT5}(\boldsymbol{x_d}) = \sum_{i=0}^{l_d/k-1} f^{sub}_{Trap-k}(\sum_{j=ki}^{ki+k-1} d_j)$, where $f^{sub}_{Trap-k} = \begin{cases} 0 & : if\ u = k \\ 1-(k-1-u)/k & : otherwise \end{cases}$

where l_c is the number of continuous variables in the function. It is a perfectly symmetric, convex and smooth function with one global optimum at $c_i = 0.0$. In our experiments the initialization range is [-10,10], unless otherwise specified. An example of the two-variable ($l_c = 2$) Sphere function is visualized in Figure 4.1.

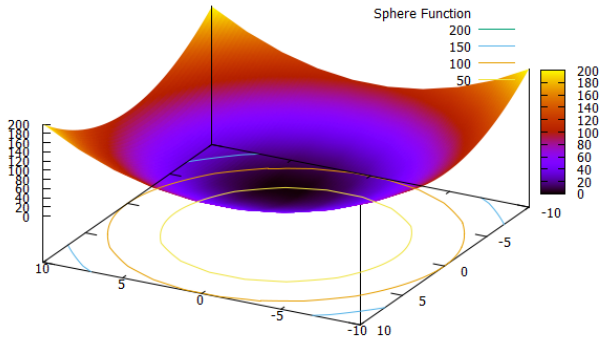


Figure 4.1: *Two-dimensional sphere function with minimum at (0,0).*

The Sphere function is considered the simplest function to optimize in the Black-Box Optimization Benchmark (BBOB) framework [42], a well-known framework used for measuring the effectiveness and performance of black-box continuous optimization algorithms.

Rotated Ellipsoid Function

Rotated Ellipsoid is considered a more complex member of the BBOB framework functions. It can be viewed as an extension of the Sphere function with two critical differences. First is a different weight factor applied in each dimension, effectively elongating the fitness contours more along each dimension with different weight. Second is a rotation factor which introduces dependencies between problem variables through a rotation matrix $\mathbf{R}(\theta)$ designed for rotation angle θ. The Rotated

Ellipsoid function is defined as follows:

$$F_{R.Ellip.}(\boldsymbol{x_c}) = F_{Ellip.}(\mathbf{R}(\theta) * \boldsymbol{x_c}) \tag{4.5}$$

where

$$F_{Ellip.}(\boldsymbol{x_c}) = \sum_{i=0}^{l_c-1} 10^{6*i/(l_c-1)} * c_i^2 \tag{4.6}$$

The $10^{6*i/(l_c-1)}$ factor is responsible for 'stretching' the function. Regarding rotation, let the rotation matrix $\mathbf{R}_{ij}(\theta)$, defined for each $i, j \in \{0, 1, ..., l_c - 1\}$, define the rotation of the plane spanned by dimensions i and j, with:

$$R_{ii}^{ij} = cos(\theta), R_{ij}^{ij} = -sin(\theta), R_{ji}^{ij} = sin(\theta) \; and \; R_{jj}^{ij} = cos(\theta) \tag{4.7}$$

The full rotation matrix $\mathbf{R}$ can then be defined as a product of all pairwise rotation matrices as follows:

$$\mathbf{R}(\theta) = \prod_{i=0}^{l_c-1} \prod_{j=i+1}^{l_c-1} R^{ij}(\theta) \tag{4.8}$$

Similarly to the Sphere function in our experiments the initialization range is [-10,10] and global optimum is located at $c_i = 0.0$. The rotation angle is $\theta = 45$.

The Rotated Ellipsoid function is known to be a difficult problem to optimize for approaches which are incapable of exploiting variable dependencies.

Onemax Function

In the discrete domain, Onemax is arguably the simplest function to optimize, exhibiting no dependencies.From a minimization perspective, it is defined as follows:

$$F_{Onemax}(\boldsymbol{x_d}) = \sum_{i=0}^{l_d-1} 1 - d_i \tag{4.9}$$

As the name suggests, to find the global optimum of Onemax, all the l_d problem variables need to be set to one. It is a well-known function often used to verify algorithmic correctness or to perform theoretical analyses on convergence properties of a given algorithm. It is however an easily solvable function, even for many less sophisticated optimization approaches such as local search.

Deceptive Trap Function

Deceptive Trap function (DT) is a well-known discrete function for which all variables are pairwise dependent [29]. The DT5 function is a binary, non-overlapping, additively decomposable composition of m of the deceptive trap functions, of length $k = 5$, where the problem length is $l_d = m * k$. From a minimization perspective, it is defined as follows:

$$F_{DT5}(\boldsymbol{x_d}) = \sum_{i=0}^{l_d/k-1} f_{Trap-k}^{sub}(\sum_{j=ki}^{ki+k-1} d_j), \tag{4.10}$$

where

$$f^{sub}_{Trap-k}(u) = \begin{cases} 0 & : if\ u = k \\ 1 - (k - 1 - u)/k & : otherwise \end{cases} \tag{4.11}$$

The global optimum of f^{sub}_{Trap-k} is achieved if all variables are set to one. A deceptive local optimum exists where all variables are set to zero. Figure 4.2 visualizes the fitness of a single five-variable block of the DT5 function. The definition of the DT function is such that the average fitness of any configuration with a certain number of ones, is lower than the average fitness of the same configurations but with zeros instead of the ones, i.e., 11*** vs 00*** with * representing a wild-card (it may be 0 or 1). Only for the full configuration, i.e. 11111 vs 00000 it is clear that ones are preferable. Because optimization with a crossover-based GA results in proliferating configurations with better average fitness, provided that crossover preserves them, it is much more likely that solutions with more zero's than ones are created and selected and thus proliferated, unless the structure of the problem is respected (i.e., entire sub functions DT5 are copied in crossover). Indeed, Thierens [86] showed that without proper crossover, a GA requires an exponentially increasing population size as the problem length of DT5 increases, whereas Harik et al [43] showed that with the proper crossover, this number scales sub-linearly.

It is important to note that trap functions can vary in the difficulty they present to the optimization algorithms. In particular, deceptiveness of a trap function is dependent on the fitness averages of the aforementioned configurations. Changing the fitness values affects the slope of the left-hand part of the graph in Figure 4.2, i.e., the slope to the local optimum. A trap function may become non-deceptive and much simpler to solve. It is moreover important to realize that, although we will encode the trap functions tightly, i.e., by using subsequent bits in the genotype bitstring, this encoding property is a priori unknown to the optimizer. Hence, independent of distributing the variables differently throughout the genotype, the optimization problem is equally complex to solve. This also means that optimization algorithms should not be configured to exploit any partial tight encoding for a fair comparison, from a BBO perspective.

4.2.2 Independently Mixed Benchmarks

With the use of the functions summarized in the previous section, we can design mixed-integer benchmarks that contain various types of intra-variable dependencies, yet do not contain any inter-variable dependencies. We consider all 4 combinations of the discrete and continuous problems where the contributions of the discrete and continuous parts are kept independent through addition, see Table 4.2.

F_1 combines the two simplest functions from each domain, the Sphere and Onemax functions, resulting in a fully independent problem. Only continuous variables are dependent in F_2, through the combination of the Rotated Ellipsoid function with Onemax. Conversely, only discrete variables are dependent in F_3 where the Sphere function is aggregated with the DT5 function. Finally, both types of intra-dependencies are present in the F_4 function, where dependencies between discrete

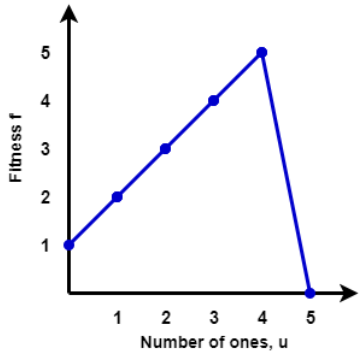


Figure 4.2: *Fitness contribution from a single five-variable block of the Deceptive Trap Function with order k=5.*

Table 4.2: $F_1 - F_4$*: Domain Independent Mixed-Integer Benchmarks*

ID	Function name	Definition
F_1	OnemaxSphere	$F_1(\boldsymbol{x_d}, \boldsymbol{x_c}) = F_{Onemax}(\boldsymbol{x_d}) + F_{Sphere}(\boldsymbol{x_c})$
F_2	Rotated Ellipsoid	$F_2(\boldsymbol{x_d}, \boldsymbol{x_c}) = F_{Onemax}(\boldsymbol{x_d}) + F_{R.Ellip.}(\boldsymbol{x_c})$
F_3	DT5Sphere	$F_3(\boldsymbol{x_d}, \boldsymbol{x_c}) = F_{DT5}(\boldsymbol{x_d}) + F_{Sphere}(\boldsymbol{x_c})$
F_4	DT5Ellipsoid	$F_4(\boldsymbol{x_d}, \boldsymbol{x_c}) = F_{DT5}(\boldsymbol{x_d}) + F_{R.Ellip.}(\boldsymbol{x_c})$

and dependencies between continuous variables exist though the combination of the Rotated Ellipsoid and the DT5 functions.

4.2.3 Cross-Domain Dependence Benchmarks

The first four of our proposed benchmark problems focus on solely the dependencies within either continuous, discrete or both parameter sub-spaces. In this section we introduce two functions that contain inter-variable dependencies.

The F_5 benchmark includes inter-variable dependencies between the continuous and discrete variables, as well as intra-variable discrete and continuous dependencies simultaneously. It is a specific combination of the previously defined F_{DT5} function and the Rotated Ellipsoid function. It is additively decomposable and consists of sub-functions pertaining to blocks of k discrete and k continuous variables.

More specifically, for a trap function with $k = 5$, there are $2^k = 32$ different binary combinations per block. A differently translated Rotated Ellipsoid function corresponds with each of those combinations (the origin of each function was randomly generated in [-5,5]). This way, the continuous function that is being optimized jointly depends on the binary counterpart, introducing non-trivial higher-order dependencies between the discrete and continuous variables that pertain to the same subset. In this benchmark the number of discrete variables is the same as the number of continuous variables: $l_d = l_c = l/2$.

$$F_5(\boldsymbol{x_d}, \boldsymbol{x_c}) = \sum_{i=0}^{0.5l/k-1}(1 + 10^a f_{sub}^{trap}(\sum_{j=ki}^{ki+k-1} d_j)) * (1 + f_{Ellipse}^{sub}(D_i^{block}, C_i^{block})), \tag{4.12}$$

where D_i^{block} is a block of five discrete variables, and C_i^{block} are the corresponding five real variables. The D block variables determine which of the 2^k different ellipsoid functions need to optimized, while the C block provides the values of the ellipsoid function variables. In order to solve this benchmark, the algorithm needs to not only solve the trap function, but also optimize the correct Rotated Ellipsoid function.

Finally, the Paired-MixDependency, or F_6 function, is a problem with mixed dependences between variable pairs, where each continuous variable is dependent on a corresponding discrete variable. In this problem, the number of discrete variables l_d and continuous variables l_c must be the same. While it contains pairwise inter-variable dependencies between corresponding discrete and continuous variable dependencies, no intra-variable dependencies exist. F_6 is defined as follows:

$$F_6(\boldsymbol{x_d}, \boldsymbol{x_c}) = 2 * \sum_{i=0}^{l_c-1}(d_i - c_i)^2 - d_i \tag{4.13}$$

It is important to note that functions $F_1 - F_4$ could be decomposed into separate problems for the discrete and continuous domains and solved independently, if perfect information was known about the problem (i.e., the problem would be treated as a white-box problem). However, this is no longer be possible for F_5 and F_6, as the inter-variable dependencies link the two sub-domains, making the these functions arguably more challenging and more representative of difficult mixed-integer problems.

4.3 Summary and Discussion

Key factors that make mixed-integer optimization especially difficult include the different types of variable dependencies that can be present in a mixed-integer problem. Problem variables may interact with each other within the same domain, either discrete or continuous (intra-dependencies). Dependencies across different domains may however exist as well (inter-dependencies). Problem composition is another key factor that affects the optimization difficulty, i.e., different proportion of discrete and continuous variables.

Based on these factors we introduced a set of customizable mixed-integer problems, which are based on well-known discrete and continuous problems. The problems introduced in this chapter serve as a foundation for performance analysis in the following chapters in terms of efficiency in discovering and processing dependencies, and scalability of any introduced algorithms in terms of resources (i.e. population size and number of evaluations) until the optimum is found (to within a sufficiently small margin).

With this, we are now ready to explore the design of efficient and effective evolutionary algorithms for optimization in the mixed-integer domain. The remainder of this thesis focuses on this domain.

5 Single-Objective GAMBIT

This chapter is based on following publications:

K. L. Sadowski, D. Thierens, and P. A. N. Bosman. GAMBIT: A Parameterless Model-Based Evolutionary Algorithm for Mixed-Integer Problems. Evolutionary Computation, 26(1):117—143, 2018.

K. L. Sadowski, P. Bosman, and D. Thierens. Learning and Exploiting Mixed Variable Dependencies with a Model-Based EA. In IEEE Congress on Evolutionary Computation (CEC'16), pages 4382—4389, 2016.

K. L. Sadowski, P. A. Bosman, and D. Thierens. A Clustering-Based Model-Building EA for Optimization Problems with Binary and Real-Valued Variables. In Proceedings of the 2015 on Genetic and Evolutionary Computation Conference, GECCO'15, pages 911—918, New York, NY, USA, 2015. ACM.

The mixed-integer domain is characterized by the presence of both discrete and continuous problem variables simultaneously. As outlined in previous chapters, one of the main challenges in designing scalable EAs is efficiently and effectively dealing with variable dependencies. In mixed-integer optimization dependencies are possible between groups of discrete problem variables as well as between groups of continuous problem variables. These are referred to as intra-variable dependencies. Additionally, cross-domain dependencies between discrete and continuous variables are possible in mixed-integer problems. We refer to such cross-dependencies as inter-variable dependencies. Such inter-variable dependencies, especially in combination with intra-variable dependencies and a black-box setting where no information about the problem is known a-priori, are key reasons why mixed-integer problems are often considered particularly difficult to solve.

Model-based Evolutionary Algorithms (EAs) have been shown to be a competent approach to solving many discrete or continuous problems, even in the presence of variable dependencies. In Chapter 2 we introduced the evolutionary algorithms LTGA and iAMaLGaM. Thanks to the ability to accurately learn and exploit problem structure, especially in the form of variable dependencies, both LTGA and

iAMaLGaM are successful algorithms for black-box optimization in the discrete and continuous domains respectively, motiving the idea that such an approach can also be applicable in the mixed-integer domain.

In this chapter we introduce GAMBIT: Genetic Algorithm for Model-Based mixed-Integer opTimization. GAMBIT is a result of our research on the applicability of model-based EAs for the mixed-integer domain. It uses as its foundation the discrete and continuous model building mechanisms of EAs, specifically LTGA and iAMaLGaM, and introduces new mechanisms that allow for effective optimization in the mixed-integer domain in case inter-variable dependencies are present. In this chapter we explain in-depth the design of GAMBIT, motivate the need and rationale behind all of its components, and verify its performance on various mixed-integer problems. Additionally, we remove the need to explicitly specify any execution parameters such as problem population size p or cluster size k through the adaptation of a parameter-less scheme.

We attempt to answer the following research questions:

- Can an integrated combination of both the discrete and continuous model-based approaches result in an efficient mixed-integer optimization algorithm?
- Can we design mechanisms to learn and effectively process inter-variable problem dependencies?
- Can we eliminate the need for specifying parameter settings of the algorithm manually without harming scalability?

The remainder of this Chapter is structured as follows. First, we introduce a general high-level summary of GAMBIT and its components. In later sections we take an in-depth look at the different components such as a balanced model integration, clustering mechanism, ability to learn and subsequently process mixed variable dependencies, achieving parameter freedom and handling constraints. We examine the performance of GAMBIT on the problem set introduced in Chapter 4 as well as on a set of constrained problems and contrast it with the performance of a selection of alternative algorithms. In a discussion section we highlight the benefits and possible downsides of our approach, and finally we summarize our findings.

5.1 GAMBIT General Overview

GAMBIT is a parameter-free model-based evolutionary algorithm designed for optimizing black-box mixed-integer problems by learning and exploiting different types of variable dependencies. To accomplish this, various mechanisms were designed and integrated.

Figure 5.2 demonstrates a top-level overview of GAMBIT and Figure 5.1 illustrates a key part of GAMBIT, its operation procedure for one cluster. An instance of the algorithm is defined by setting two key parameters: the population size n and the number of clusters k. Both of these parameters are however handled by a parameter-free scheme, explained in detail further in this chapter, allowing the parameter-free

version of GAMBIT to be executable "out-of-the-box" without the need of the user to specify any parameters.

GAMBIT's optimization capabilities come from its capacity to capture and exploit problem structure through estimation of important variable dependencies. The ability to learn over time which these dependencies are allows for creation of variable subsets which represent important building blocks of the problem. Once such building blocks are established, they can be used to exploit linkage problem structure and generate better solutions more effectively. This approach of exploiting a problem's linkage structure works excellently in LTGA, where a similar methodology is in place for discrete variable optimization. Mixed-integer optimization introduces new optimization challenges, however. The existence of discrete and continuous variables creates a potential for intra- and inter-variable dependencies to exist that LTGA alone is not equipped to handle. GAMBIT makes use of mechanisms based on LTGA and iAMaLGaM for handling discrete and continuous variables respectively. It further introduces a new way to learn and process mixed inter-variable dependencies, as well as an efficient and balanced mechanism to integrate each of these components during optimization.

The core functionality of GAMBIT can be summarized as follows. A population is first initialized with random values for each variable (if known, initialization ranges can be defined for each variable). At the beginning of each generation this population is clustered into k equally sized clusters, based on the proximity of each solution to k solutions taken from the population which are furthest apart from each other according to a mixed distance described in detail in Section 5.2.3. For the duration of the generation, each of these sub-populations P_k are processed individually with an integrated-models mechanism. The integrated-models mechanism learns and attempts to exploit the structure of a given problem by considering three types of possible variable dependency types: discrete, continuous (intra-dependencies) and mixed discrete-continuous (inter-dependencies). A family-of-subsets structure is created containing discrete, continuous and mixed subsets of problem variables, which represent important solution blocks. The linkage sets in the FOS are then used to sample the offspring solutions, following a balanced-sampling mechanism explained in detail further in this chapter. From each of the k clusters, an integrated-models mechanism generates an equal number of offspring solutions which are joined and form the offspring population to be used for the next generation. This process continues until a specified termination criterion is met, based on reaching a specified runtime maximum, number of allowed evaluations, or finding the globally optimal solution (if known). Thanks to the adaptation of a parameter-free scheme, the key initialization parameters, population size and number of clusters, are internally managed and do not require specification by the user, making GAMBIT a truly "out-of-the-box" black-box mixed integer optimization algorithm.

5.2 GAMBIT Details on Components

GAMBIT can be viewed as an integrated collection of components which together form a novel optimization algorithm, where four major components can be identified

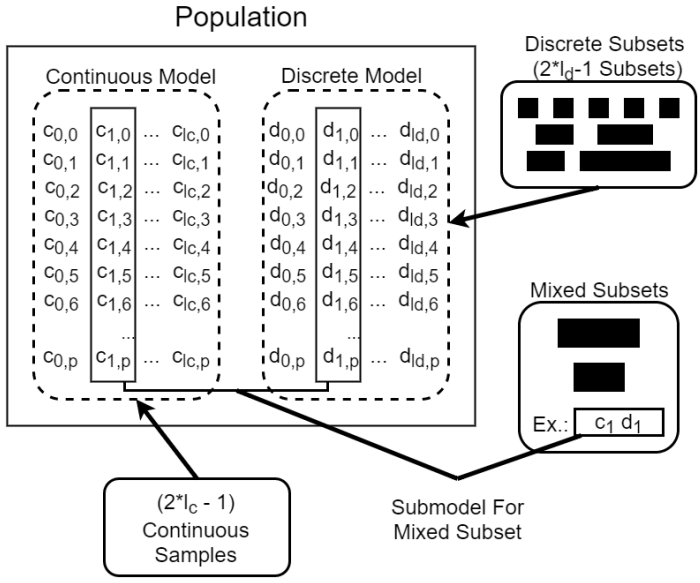


Figure 5.1: *GAMBIT and the integrated model mechanism for a single cluster*

GAMBIT Overview

$\mathcal{P} \leftarrow$ GENERATEANDEVALRANDOMPOPULATION(n)
while $\neg$TERMINATIONCRITERIONSATISFIED **do**
  $\mathcal{C} \leftarrow$ DETERMINECLUSTERCENTERS$(\mathcal{P}, k)$
  for $j \in \{0, 1, \ldots, k\}$ **do**
    $\mathcal{P}_k \leftarrow$ GROWSUBPOPFROMCENTERS$(\mathcal{P}, \mathcal{C}_k)$
    $\mathcal{O}'_k \leftarrow$ INTEGRATEDMODELSMECHANISM$(\mathcal{P}_k)$
  $\mathcal{P} \leftarrow \mathcal{O}'_0 \cup \mathcal{O}'_1 ... \cup \mathcal{O}'_k$

Figure 5.2: *Pseudo-code overview of GAMBIT. $\mathcal{P}$ represents the population of size n. $\mathcal{C}$ is a set of solutions representing k cluster (also called sub-population) centers. $\mathcal{P}_i$ represents the i-th sub-population and $\mathcal{O}'_i$ is the offspring generated from sub-population i.*

that are responsible for dealing with different challenges inherent to mixed-integer optimization. Each of the key components of GAMBIT is detailed in this section, along with the reasoning behind and the importance of these components.

First, a core functionality of GAMBIT relies on the ability to estimate intra- and inter-variable dependencies in order to effectively construct new solutions. To this end, an integrated models mechanism adapts and extends the family-of-subsets (FOS) functionality already present in LTGA to allow the existence of purely discrete, purely continuous and mixed discrete-continuous subsets of problem variables. Second, the integrated models mechanism should also exploit the learned dependencies appropriately. This entails sampling appropriately according to the type of modeled variables but also balancing the model learning, updating, and sampling solutions.

Third, GAMBIT consists of multiple instances of the integrated models mechanism. A clustering mechanism partitions the population and assigns which sub-populations are processed with the different integrated models mechanism instances. Finally, fourth, parameter settings are handled internally with a parameter-free scheme.

5.2.1 Integrated Models Mechanism: Learning and Representing Variable Dependencies

In order to deal with different types of dependencies, GAMBIT includes a different mechanism for each type. These mechanisms result in generating three types of variable subsets: exclusively discrete, exclusively continuous and mixed, which form family-of-subsets that can be used to generate new offspring solutions. In this section we focus on explaining how different variable dependencies are estimated and how the FOS structure is populated with them. While intra-variable dependency learning and processing is inherited from LTGA and iAMaLGaM, inter-variable dependency learning requires an additional mechanism.

Intra-variable Dependencies

Following the LTGA procedure detailed in Chapter 2, GAMBIT uses a discrete mutual information calculation to generate a linkage tree containing subsets of discrete problem variables. These $2l_d - 2$ subsets populate the FOS structure in LTGA. The same methodology is used with GAMBIT, with two notable differences. First, GAMBIT allows the subset at the root of the linkage tree to be included in the FOS structure, making the number of discrete subsets in the FOS $2l_d - 1$. LTGA ignores this final subset as it represents all of the problems variables. However, with GAMBIT all of the discrete variables are still only a subset of all the problem variables as continuous variables are also present in the problem, making this additional subset potentially useful during optimization. Second, these discrete subsets make up only part of the entire FOS structure, as continuous and mixed subsets are also possible in GAMBIT.

While LTGA relies on mixing discrete subsets to create offspring, iAMaLGaM generates the continuous variables at once, through building, maintaining and sampling a Gaussian distribution over all variables. In order to retain this functionality, while at the same time incorporate it into GAMBIT's FOS structure, we introduce continuous subsets to the FOS. Unlike the discrete subsets however, which can represent any sub-structure of the discrete variables, the continuous subsets in GAMBIT include all of the continuous variables in each subset so as to capture all possible dependencies between all continuous variables. This means that while each of the discrete subsets is unique in terms of composition and varies in length, every continuous subset is identical.

The number of discrete subsets in each FOS is $2l_d - 1$. In order to establish a balanced FOS, $2l_c - 1$ continuous subsets are added to the FOS, each containing all the l_c continuous variables. This is important because through optimal mixing (in LTGA), many small improvement steps may be made. Adding only a single

full continuous FOS element could very well cause the discrete part of the solutions to converge much faster, potentially eliminating much needed diversity before the impact of optimizing for the continuous variables can be observed, which may well influence which discrete updates are really improvements.

Inter-variable Dependencies

In order to model inter-variable dependencies within a FOS, allowing subsets that contain both discrete and continuous variables is necessary. Estimating such dependencies is not straightforward and requires the ability to quantify some measure of strength of dependency between variables of different types.

Mutual Information is a very useful measure of variable interactions [27] [52]. While the discrete subset estimation of GAMBIT already uses mutual information to generate subsets with a linkage-tree, the same calculation is not directly applicable to the mixed-variable case, as mutual information as calculated in the discrete and in the continuous domains are not directly comparable.

Because of this, we utilize a recently introduced approach called the nearest neighbor method [73]. This method relies on calculating the Jensen-Shannon divergence, which measures the dissimilarity between two or more continuous probability distributions [53] [58] [38].

The nearest neighbor method aims to determine if different values of the discrete variable X are biased towards different values of the continuous variable Y. More specifically, for each data point (x_i, y_i) a number I_i is computed based on the nearest neighbors of (x_i, y_i) considering only the continuous variable y_i. First, the k-th closest neighbor of point (x_i, y_i) is found whose value of the discrete variable is x_i, i.e. point (x_i, y_j) for which Euclidean distance $d = |y_i - y_j|$ is the k-th smallest. Using d as the distance to the k-th neighbor, the number of neighboring points m_i that are within distance d, is counted from the entire data set, where distances again only consider real-valued variables, but now the discrete value is not restricted to match x_i. We compute

$$I_i = \psi(N) - \psi(N_{xi}) + \psi(k) - \psi(m_i) \tag{5.1}$$

where ψ is the digamma function, N is the number of all data points, N_{x_i} is the number of data points whose discrete values equal x_i. Now, the estimation of the MI between the discrete-continuous pair (X,Y) is an average I_i over all the data points:

$$I(X, Y) = \langle I_i \rangle = \psi(N) - \langle \psi(N_x) \rangle + \psi(k) - \langle \psi(m) \rangle \tag{5.2}$$

In our implementation we use $k = 3$, following a recommended guideline for this method [73].

With the use of the nearest neighbor method, GAMBIT is able to calculate a mutual information-based measure of variable dependencies between discrete and continuous variables. Motivated by the success of the discrete linkage-tree building procedure used for generating discrete subsets, we now propose to obtain the mixed subsets in a similar fashion. Specifically, a second linkage-tree spanning across all $l_c + l_d$ problem variables is obtained using the nearest-neighbor based mutual information. For this, any mutual information value between subsets of exclusively

the same types of variables are set to zero, while the remaining values are calculated using the nearest-neighbor method. The procedure results in a tree formation based on exclusively mixed dependency estimation. This mixed-variable linkage tree generates $2(l_c + l_d) - 1$ linkage-tree nodes. All the leaf nodes can be discarded as they represent singleton nodes of either discrete or continuous variables. The root node is just a representation of all the problem variables and can also be discarded. The remaining $l_c + l_d - 2$ nodes represent mixed-variable dependencies, and are added to the FOS alongside of the exclusively discrete and continuous subsets. The FOS formation used with GAMBIT, based on the two linkage-tree approach, is visualized with a simple example in Figure 5.3.

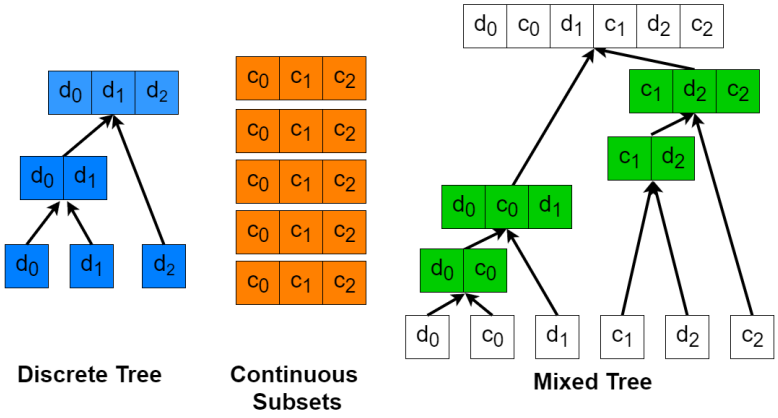


Figure 5.3: *Example of generating the FOS subsets with GAMBIT with 3 different mechanisms for discrete, continuous and mixed variables. According to GAMBITs procedure the FOS will be populated with $2l_d - 1$ discrete subsets (blue), $2l_c - 1$ continuous subsets (orange) and $l_d + l_c - 2$ mixed subsets (green).*

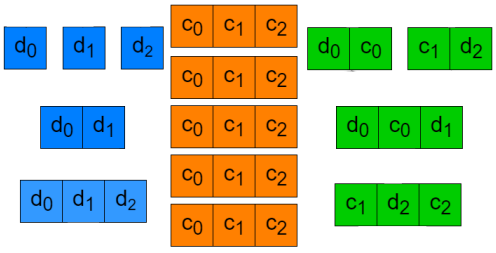


Figure 5.4: *Example FOS structure of GAMBIT for 3 discrete and 3 continuous variables, which includes discrete, continuous and mixed subsets.*

5.2.2 Integrated Models Mechanism: Processing Dependencies and Sampling Offspring Solutions

Integrative dependency processing can be considered key to the success of an approach using model-based algorithms to solve mixed-integer problems in case non-trivial mixed dependencies exist. A crucial aspect is determining a way of maintaining a balance between structure learning and offspring creation done by sampling from a discrete model when a discrete subset is encountered, sampling from the continuous model when a continuous subset is encountered, and sampling from a mixed model when a mixed-subset is encountered. Because of the nature of mixed-integer landscapes, performing independent learning with different models for the continuous and discrete variables can easily lead to premature convergence when the possible values for some variables are not sufficiently explored, or to over-exploration when too much focus is given to some variables. For instance, having optimization of only one type of variable proceed much faster (i.e. faster convergence) may lead to too much diversity reduction on the side of the other type of variable for efficient optimization of the entire solution to be realized. This potentially even increases the chances of getting stuck in local optima in the space of the problem variable type where search progresses much slower and is thus, via selection being biased to improvements, dominated by the search on the variable type that is processing faster, being forced to discard values that happen to be associated with values in the same solution for the other types of variables that result in high fitness. This problem is aggravated if underlying dependencies exist between real and discrete domains.

With a FOS structure populated with discrete, continuous and mixed subsets representing important variable groupings as illustrated in Figure 5.4, GAMBIT can now utilize them to sample and create new solutions. The integrated models mechanism handles each subset type differently.

The details of the integrated models mechanism insofar learning is concerned have already been presented in previous chapters. In Figure 5.5 these details are complemented with details on the use of these models for generating new solutions and are explained here. The initial population is randomly generated. Each solution $\mathcal{P}_i$ in the population consists of a continuous component $\mathcal{P}_{i_c}$ as well as a discrete component $\mathcal{P}_{i_d}$. At the beginning of each generation, a clustering mechanism is used to group solutions, on the population level, in k clusters. This mechanism is detailed in the next section. For each of these clusters the discrete model is updated and a linkage tree is built over all discrete variables. It is used to generate the $2l_d-1$ discrete FOS elements as previously illustrated in Figure 2.4. The FOS is then extended with $2l_c - 1$ continuous subsets, each containing all l_c continuous variables. Finally $l_c + l_d - 2$ mixed-subsets are generated in the manner explained in the previous section. The FOS subsets are then exhaustively processed in a randomized order. For each FOS subset, each individual in the cluster is subjected to variation. The processing of the subsets depends on the subset type.

If the subset type is continuous, the continuous model is first updated. Each offspring solution is created by copying all discrete variables from the current parent solution, and sampling all new continuous variables. The offspring solution is always accepted, following the iAMaLGaM sampling procedure. Updating the con-

tinuous model consists of re-building a covariance matrix, based on $\tau = 0.35$ fraction of best solutions in the cluster, and updating iAMaLGaM's intergenerational mechanisms.

If the subset is discrete, the variables specified by the subset are copied from a donor solution randomly selected from the current cluster, and copied onto the current solution, while the remaining discrete variables and all of the continuous variables remain unchanged. This offspring solution is only accepted if it is not worse than the parent solution. This procedure is illustrated in Figure 2.5.

When a mixed FOS subset is encountered, sub-models are created. Specifically, the sub-population that is currently handled by the integrated models mechanism, is grouped into model clusters, this time on the population cluster level, however now only considering the variables present in the mixed subset for distance calculation. This results in k equally-sized sub-clusters for every mixed subset in the FOS. The value for k is set equal to the number of clusters used by the main (i.e., population) clustering mechanism of GAMBIT. This way k does not need to be explicitly specified. A sub-model represents only the variables present in those linkage sets, making it relatively inexpensive in terms of additional computing resources needed (i.e., computing covariance matrices and decomposing them for sampling). When a mixed subset is encountered, for each solution a donor solution is randomly selected from the cluster. The sub-cluster to which this donor belongs, is probabilistically calculated (i.e., maximum-likelihood match). The discrete subset variables are copied from the donor, into the current solution. The continuous variables are sampled from the sub-model which was matched to the mixed subset. Only after both, the continuous and discrete variables of the mixed subset have been sampled, changes are evaluated and kept only if it results in an improvement. Otherwise, the change is rejected. Pseudo-code for this procedure is given in Figure 5.5.

After every subset is processed for every solution, the generation ends. The offspring solutions generated by each instance of the integrated models mechanism are merged together, forming a new population, and is used for clustering in the next generation.

5.2.3 Clustering Mechanism

Clustering offers a means of partitioning the search space and allowing an EA to focus its search on different regions of the search space. A well-designed clustering approach can be especially useful in the black-box setting, where the problem structure is initially unknown and may include features such as many local optima.

A clustering mechanism is one of the key components of GAMBIT. The clustering mechanism offers the ability to partition the problem population on the solution level, resulting in sub-population groupings that are then optimized with an instance of the integrated models mechanism.

The overview of this clustering mechanism is as follows. A constant number k of clusters is predetermined. A population is generated randomly with n solutions that are evaluated. This population is then clustered into k equally-sized (n/k) clusters. A separate instance of the integrated models mechanism is then associated with each cluster. This is, within each cluster, the variable linkages and Gaussian-distribution

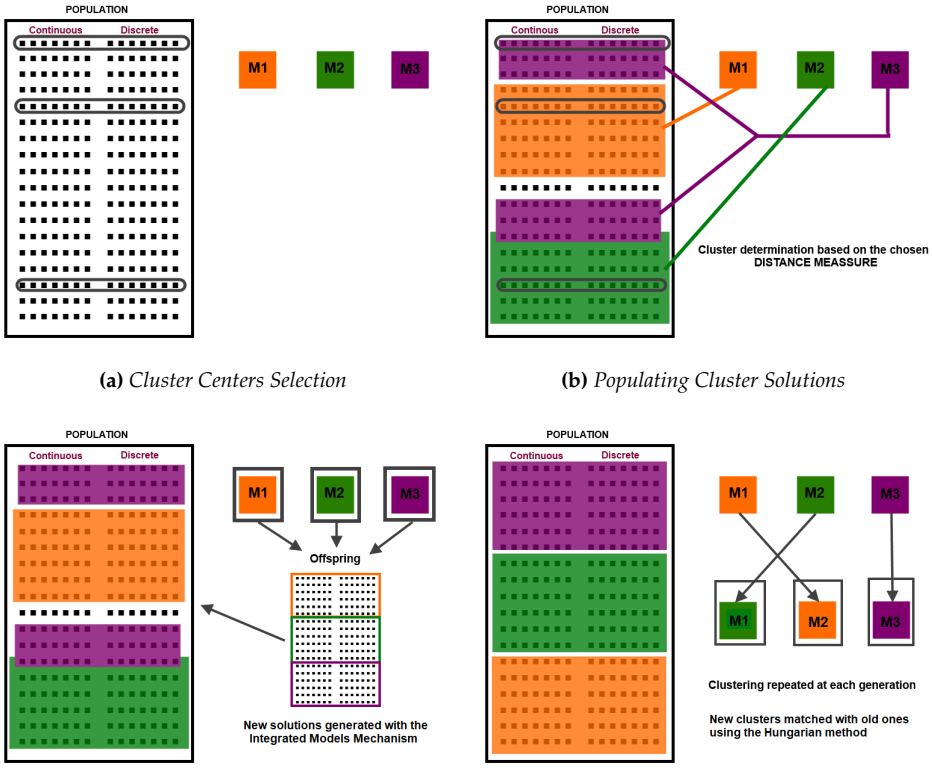
Integrated Model Sampling for a cluster sub-population $\mathcal{P}_k$

```
INTEGRATEDMODELMECHANISM(P_k)
  F = LearnFOS(P_k)
  for i ∈ {0, 1, ..., (|F| − 1)} do
    if F^i is all continuous then
      S ← TRUNCATIONSELECTION(P_k, τ)
      UPDATEGAUSSIANPARAMETERS(k, S)
      for j ∈ {0, 1, ..., n − 1} do
        ((P_k)_j)_{F^i} ← SAMPLECONTINUOUSMODEL(k, F^i)
        EVALUATEFITNESS((P_k)_j)
    if F^i is all discrete then
      for j ∈ {0, 1, ..., n − 1} do
        O ← (P_k)_j
        donor ← GETRANDOMSOLUTION(P_k)
        (O)_{F^i} ← (donor)_{F^i}
        EVALUATEFITNESS(O)
        if fitness(O) at least as good as fitness((P_k)_j) then
          ((P_k)_j)_{F^i} ← (O)_{F^i}
    if F^i is mixed then
      P_{k_sub}[0, ..., k − 1] ← GROUPINTOCLUSTERS(i, k, P_k)
      UPDATEMIXEDSUBMODELS(i, P_{k_sub})
      for j ∈ {0, 1, ..., n − 1} do
        SubID ← DETERMINESUBPOPULATION(j)
        O ← (P_k)_j
        (O)_{F^i} ← SAMPLESUBMODEL(i, P_{k_sub}[SubID])
        EVALUATEFITNESS(O)
        if fitness(O) at least as good as fitness((P_k)_j) then
          ((P_k)_j)_{F^i} ← (O_k)_{F^i}
  return P_k
```

Figure 5.5: *Pseudo-code for generating solutions for mixed integer problems with GAMBITs Learning Models.*

parameters for the discrete and continuous variables are learned respectively. Based on the learned information, each model instance generates n/k new solutions, which are added into the main population. The clustering process begins anew in every generation. After clustering takes place, the clusters from the previous generation are matched with the newly created clusters using the distance-minimizing Hungarian method [54], where the cluster distance is the sum of all pairwise distances between solutions. Intuitively, this approach identifies the pairing of solutions that has the smallest sum of paired distances. The distance metric used between solutions is the same as the one used for clustering. By doing so, the intergenerational mechanisms needed by iAMaLGaM can proceed smoothly as the clusters do not chaotically change their positions over generations. The clustering process is summarized in pseudo-code form in Figure 5.7 and an example visualization is provided in Figure 5.6.

The clustering process itself takes place at the beginning of every generation. First, k cluster leaders are selected. The best solution available in the population is



(a) *Cluster Centers Selection*

(b) *Populating Cluster Solutions*

(c) *Generating Offspring*

(d) *Matching IMM Instances*

Figure 5.6: *A visualization of GAMBIT's Clustering Mechanism and integrated models mechanism during a generation for cluster size $k = 3$.*

always selected as a cluster center. Other centers are iteratively chosen as solutions with the largest minimal distance to any already chosen center. Once the centers are determined, n/k solutions closest to the chosen centers are added to the clusters in accordance with the chosen distance metric. This approach also allows certain solutions to be part of multiple clusters. This cluster selection process is based on a similar approach used for Multi-Objective optimization first presented in [13].

```
Clustering-Based Model-Building EA
for i ∈ {0, 1, ..., n − 1} do
  P_i ← CreateRandomSolution()
  EvaluateFitness(P_i)
t ← 0
while ¬TerminationCriterionSatisfied do
    C(t) ← PopulateClusters(P)
    if t > 0
        MatchClusters(C(t), C(t − 1))
    for j ∈ {0, 1, ..., k − 1} do
        P_i ← ClusterBasedVariation(C_i)
    P ← P_0 ∪ P_1 ∪ ... ∪ P_{k−1}
    t ← t + 1
```

Figure 5.7: *Pseudo-Code overview of the Clustering-based mechanism*

A final but important component of the clustering mechanism is the distance metric used for calculating the distance between two mixed-integer solutions. In order to consider both discrete and continuous variables together, a joint distance metric is required. In GAMBIT, this is accomplished by considering discrete and continuous variables together in the continuous domain. The continuous variables are normalized and scaled down to a 0-1 range where the current largest value in the population takes on the value of one, and the smallest zero. Binary variables are then treated as real values and the Euclidean distance is used. While other distance metrics were considered and tested, they resulted in less efficient optimization by GAMBIT.

5.2.4 Parameter Freedom

When experimental results of running algorithms are presented, they are often associated with certain parameter settings that were used. For some parameter settings a given algorithm can perform much better than for others on a given problem. As indicated earlier, GAMBIT requires two crucial parameters: population size p and number of clusters used k. In this section we explain why the need for setting these parameters can be problematic, especially in a black-box optimization setting, and propose a methodology that allows for a parameter-free execution of GAMBIT.

The population size is an essential parameter of any evolutionary algorithm. Determining a population size that results in efficient performance, however, is often difficult. Setting the population size too small will result in premature convergence and inability to solve the problem. Choosing the population size too large may re-

sult in an unnecessary overhead leading to inefficient performance in terms of evaluations needed. In research on population-based algorithms, very often results are reported based on empirical data obtained using the optimal, or at least to some degree tuned, population size for a given problem. While the use of bisection to find the optimal population size is very useful to examine algorithmic performance from the fundamental perspective, it is very often infeasible and not useful in practice. For example, if the optimum of a problem is not known, it may be impossible to know if a given population size results in good or poor performance. Additionally, once a problem is solved, it makes no sense to try solving the problem again for a smaller population size.

A priori predicting a good population size for a given problem in a black-box setting can be near impossible. While it is generally true that when the dimensionality of a problem increases, the population size required to solve it also increases, there is very little information that tells us what this optimal population size might be for different problems. This is because many factors affect problem difficulty such as multi-modality, variable interactions or constraints.

Similarly to the population size parameter, determining an optimal number of clusters for a given problem prior to execution may often be impossible in practice. Choosing the wrong number of clusters however, can have a negative impact on the performance. An insufficient number of clusters may lead to inability to exploit more complex problem structures, while too many clusters will likely result in significant overhead.

In order to design an algorithm that may be useful in a real-world setting, the problem of determining values for the population size and the number of clusters needs to be addressed. We hereto adapt the Parameter-Free scheme initially proposed for discrete optimization by [44]. Our goal is to determine if it is possible to adapt this scheme to GAMBIT while retaining similar performance in terms of scalability as with the optimal population size. To this end, we will test the performance, with and without the parameter-free scheme, of GAMBIT on problems with different landscape features and with different problem composition.

Methodology

The population-free scheme was initially proposed for discrete optimization. In recent work this approach was successfully used with GOMEA, and was called an interleaved multi-start scheme (IMS) ([63] [64]). In this scheme, the optimization algorithm is initialized dynamically over time for different population sizes. The generational progress of each instance is inversely proportional to the instance's population size, allowing for an evaluation balance between instances. Additional mechanisms exist which determine when to terminate smaller instances based on their relative performance or when to start new, larger instances. The general idea behind this scheme is to give instances with smaller population sizes a chance to solve a problem, but allow larger instances to attempt to solve it simultaneously without the need of instances with smaller population sizes to reach a termination criterion first.

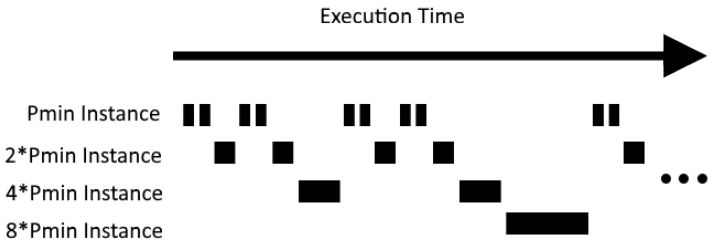


Figure 5.8: *A visualization of the population size free scheme for $b = 2$.*

More specifically, an instance of an algorithm is first created for a certain minimal population size. This instance runs for b generations, where b represents the base, suggested to be 2 or 4. After this, a new instance of the algorithm is created, with a doubled population size. This instance will perform a single generational step at the frequency $1/b$ smaller than the previous instance. After b generational steps of this instance (by which time the first instance made b^2 generational steps), again a new instance with a double population size is created and the process continues. An visualization of this process is provided in Figure 5.8.

A major drawback of using an iteration rather than an interleaved scheme is the need to specify when the algorithm should move on to a larger population size. Such stopping criteria are very often difficult, if not impossible, to specify. With the interleaved approach, this is no longer necessary as algorithm instances with larger population sizes will be utilized, alongside of smaller population size instances.

Some modifications are made to this scheme. If for a given GAMBIT instance, the average fitness of solutions selected by GAMBIT is higher than the average fitness of the selected solutions for an instance that is still running with a smaller population size, the instance with the smaller population size as well as all instances with a smaller population size than the one that is being outperformed, cease execution. This is a way to save needless evaluations, since a larger instance already reached better fitness levels. Additionally, the best current solution out of any instance is always preserved and shared among all instances. In our implementation we use b = 4. The minimal, or starting population size is based on iAMaLGaM's population size guidelines, which is $10\sqrt{l_c}$. We adapt this guideline to the mixed-integer case, by setting the initial population size to $10\sqrt{l_c + l_d}$ to consider the addition of variables from the discrete domain.

Determining the number of clusters is done in a similar manner. Every time a new instance (with a doubled population size) is introduced, the number of clusters for this instance is increased by one from the previous instance. With a linearly growing number of clusters k, and exponentially growing population size n, the number of solutions in each cluster will also increase with time, as the number of population members per cluster is n/k. In our implementation the starting cluster size is $k = 1$.

5.3 Experiments

Good performance of model-based EAs, LTGA and iAMaLGaM in their respective domains does not necessarily translate to these EAs having good performance in the mixed-integer space. To see whether an EA such as GAMBIT with its dedicated mechanisms truly has additional value, we will compare it to several other approaches that leverage LTGA and iAMaLGaM in more straightforward ways. An empirical analysis is performed with the following key considerations:

- **Problem composition**: how does GAMBIT behave when the problem consists of different fractions of continuous and discrete variables under different types of dependencies present?
- **Simple approaches**: what is the difference in performance when we utilize already successful algorithms for either discrete or continuous domain in the mixed-integer domain directly?
- **Inter-variable dependencies**: is using mechanisms specifically designed for mixed-dependencies worth the overhead associated with them?
- **Alternative algorithms**: how do state-of-the-art commercial and non-commercial alternative algorithms compare on our benchmarks?

In order to examine these considerations, this section is organized as a series of subsections. Each subsection consists of the description and motivation behind the specific consideration as well as additional information about algorithms or problems specific to the given consideration. Finally, each of the subsections concludes with a description and analysis of the results pertaining to the experiments performed for the specific consideration.

5.3.1 Problem composition

Problem composition in terms of the number of variables of a different type is a characteristic inherent to mixed-integer optimization, and not present in exclusively discrete or continuous optimization. We wish to examine the behavior of GAMBIT when the problem composition is even, when it is mostly discrete and when it is mostly continuous. We hope to retain a favorable scalability in each of these cases, just as iAMaLGaM and LTGA are capable of doing in their respective domains. Furthermore, we hope to show that the adaptation of the parameter-free scheme does not introduce a substantial increase in terms of evaluations needed.

We compare the parameter-free GAMBIT with GAMBIT parameterized with the optimal population size determined via bisection aimed at finding the population size that leads to requiring the smallest number of evaluations. Specifically, we use bisection to find the population size that corresponds to the minimal total evaluations needed to solve a problem with the precision of 10^{-10} at least 29/30 times. We test benchmarks with different landscape characteristics: F_1, F_2, F_3 and F_4 (see Chapter 4). For each of the problems, different problem sizes and problem compositions are considered. Let ρ describe the ratio $\rho = l_c/(l_c + l_d)$ of the continuous

variables in a given problem. For instance $\rho = 1.0$ denotes a problem where all variables are continuous, while $\rho = 0.5$ represents a problem with the same number of discrete and continuous variables. We consider different values for ρ: 1.0, 0.75, 0.5, 0.25, 0.0. The success criterion is finding the optimum in at least 29 out of 30 runs. All benchmark problems have an optimal value of 0. However, since part of these problems is continuous, we consider a benchmark problem solved if the value-to-reach of 10^{-10} is reached.

Heat-maps in Figure 5.9 show how many evaluations are needed for problems with different dimensionality and variable composition. As could be expected, as the problem dimensionality increases, so does the number of evaluations required. Additionally, a larger number of evaluations is also needed when the problem composition becomes more continuous (larger values of ρ), or when problems containing discrete dependencies (F_3 and F_4) are present.

Intuitively, F_1 should be the simplest problem, as it contains no parameter dependencies. As the problem composition shifts towards a more continuous landscape, GAMBIT requires more evaluations. The required population size is less affected by the problem composition for F_1 than for the remaining benchmarks. This can be explained by the simplicity and independence of all problem variables.

The effects of changing the problem composition strengthen when intra-dependencies are introduced into the problem landscape. As with F_1, for the remaining benchmarks more evaluations are also required for the same problem sizes as the composition of the problem shifts towards larger numbers of continuous variables. Moreover, as expected, benchmarks which contain dependencies within the continuous sub-space, F_2 and F_4, require larger number of evaluations than F_1 or F_3.

This shows that in addition to problem length, the composition of the problem and variable dependencies are a big factor for efficiency in terms of evaluations and population sizes. From the heat-maps it follows that the parameter-free GAMBIT obtains a fairly similar behavior as the GAMBIT instance with an optimal population size in terms of evaluation requirements as a function of problem size and composition. We explore the overhead in more detail by considering the scalability of the GAMBIT equipped with the parameter-free scheme.

Scalability

Figure 5.10 demonstrates changes in scalability of population size and evaluations over benchmarks $F_1 - F_4$ when the proportions of discrete to continuous variables is changed. Results are shown on a log-log scale. This means that polynomial scalability is indicated by straight lines. From the scalability graphs it is clear that factors such as variable ratios and dependencies can affect the behavior of GAMBIT. GAMBIT does exhibit polynomial scalability on the tested mixed-integer problems, which shows that when faced with different mixed-integer problem compositions that include intra-domain dependencies GAMBIT retains the efficient scalability of LTGA and iAMaLGaM. Comparison between the optimal population size and the parameter-free scheme equipped GAMBIT version is presented in Table 5.1.

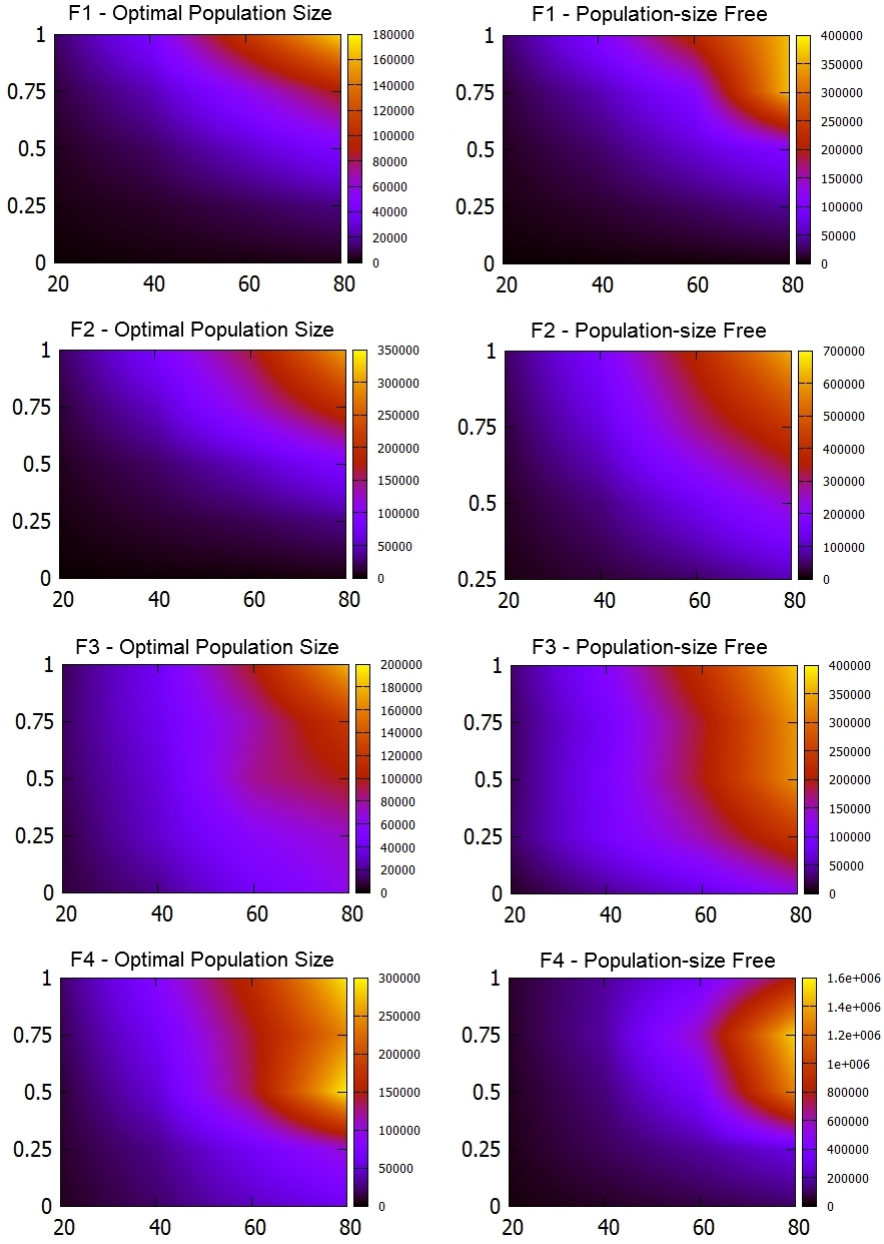


Figure 5.9: *Heat maps representing the number of evaluations needed with the optimal population size (left column) and with the parameter-free scheme (right column). Horizontal axis: problem length, the vertical axis: fraction of continuous variables ($l_c/(l_c + l_d)$)) in the problem.*

Table 5.1: *Regression coefficients for scalability of evaluations on F_1 through F_4 benchmarks. α_ρ is the regression coefficient where ρ is the fraction of continuous variables in the problem.*

Population Scheme		Scalability Coefficient				
		α_0	$\alpha_{0.25}$	$\alpha_{0.5}$	$\alpha_{0.75}$	$\alpha_{1.0}$
F1	Optimal Population-Size	1.5	1.5	1.8	2.0	1.9
	Parameter-Free	1.5	1.6	1.8	2.2	1.9
F2	Optimal Population-Size	1.5	1.6	1.8	1.9	1.9
	Parameter-Free	1.4	1.7	2.0	2.0	1.9
F3	Optimal Population-Size	1.4	1.7	1.8	1.9	1.9
	Parameter-Free	1.7	1.6	1.9	1.9	1.8
F4	Optimal Population-Size	1.6	1.8	2.4	2.2	1.9
	Parameter-Free	1.6	1.7	2.5	2.6	2.0

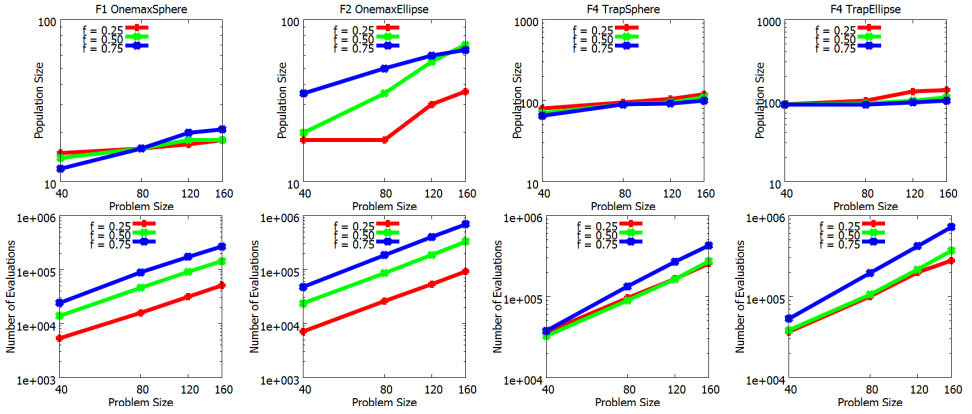


Figure 5.10: *Scalability of population size and evaluations required by GAMBIT for benchmarks $F_1 - F_4$. Fraction f is the fraction of continuous variables $(l_c/(l_c + l_d))$ in the problem.*

Specifically, Table 5.1 provides the regression coefficients for scalability in terms of the number of evaluations. The results show that performance in terms of evaluations scales more favorably as the problem shifts towards having relatively more discrete variables. However, on all the tested configurations GAMBIT achieves a polynomial scalability in terms of number of evaluations needed, and therefore is capable of retaining some of the excellent performance in terms of scalability of LTGA and iAMaLGaM in the mixed-integer domain.

Additionally, data provided in Table 5.1 suggests that GAMBIT with the population-free scheme can achieve very similar scalability as the scalability attained with an optimal population size. In some cases (e.g. $F_4 : \rho = 0.75$) the scalability with optimal population size is slightly better, in the majority of cases the population-free scheme scales just as good. In some cases (e.g., $F_2 : \rho = 0$) the parameter-free scheme even allows for slightly better scaling.

5.3.2 Simple Approaches

In this section, we explore the idea to adapt and expand the LGTA and the iAMaLGaM algorithms from the discrete and continuous domains respectively for the mixed-integer domain in the most straightforward manner. A natural question then arises: can the model-based abilities of these algorithms be easily individually extended to mixed integer problems in a way which retains their effectiveness or at least be as effective as the more elaborate design behind GAMBIT?

MI-LTGA

In MI-LTGA a mixed-integer problem is treated as if all variables were discrete. To this end, a discretization of the continuous variables takes place, for which the well-known Gray encoding ([33], [66]) is used. With Gray coding adjacent numbers always differ by exactly one bit. This solves the problem of Hamming cliffs which can occur with decimal encoding, where a small change to the encoded number can be mapped to a very large difference in representation in the binary space. We use a base of $b = 24$ bits to represent every continuous variable, in order to have sufficient precision. This means that a mixed-integer problem of size length $l = l_d + l_c$ where l_d and l_c represent the number of discrete and continuous problem variables respectively becomes a fully discrete problem with problem length $l_d + b * l_c$.

MI-iAMaLGaM and MI-iAMaLGaM-LS

MI-iAMaLGaM is the continuous analogy of MI-LTGA. All binary variables are initialized to either 0 or 1, but are otherwise treated as if they were real-valued. These variables are converted to binary values during function evaluation (i.e., the solution itself is not changed), where values smaller than 0.5 are considered to be 0, and values greater or equal to 0.5 are considered to be 1.

Additionally we consider a variant of MI-iAMaLGaM called MI-iAMaLGaM-LS in which a local search (LS) variant is introduced. Local search is an optimization heuristic which repeatedly performs small step size changes to a given solution, until a local minimum is found. Local searches are often used in the discrete domain where the step size is a single bit flip. This functionality allows the continuous model-building and sampling of iAMaLGaM to guide the search, while a first-improvement local search hill-climbs in the discrete variable space. The use of local search in a black-box algorithm is potentially expensive, as each bit flip requires a full function evaluation. The benefits however include making sure the discrete part of the solution is always at a local optimum, which may in turn simplify the optimization process for iAMaLGaM.

Results

In order to examine how the simple approaches perform, we compare their performance on functions F_1 through F_3 where different types of variable dependencies are present. For contrast, we also report the performance of GAMBIT.

The experimental setup is as follows. For each algorithm we use bisection to find the population size that corresponds to the minimal total evaluations needed to solve a problem with the precision of 10^{-10} at least 29/30 times. Only the evaluations used for the optimal population size of each algorithm are presented, as well as the corresponding population sizes. Results on functions $F_1 - F_3$, where $l_c = l_d = l/2$, are presented in Figure 5.11.

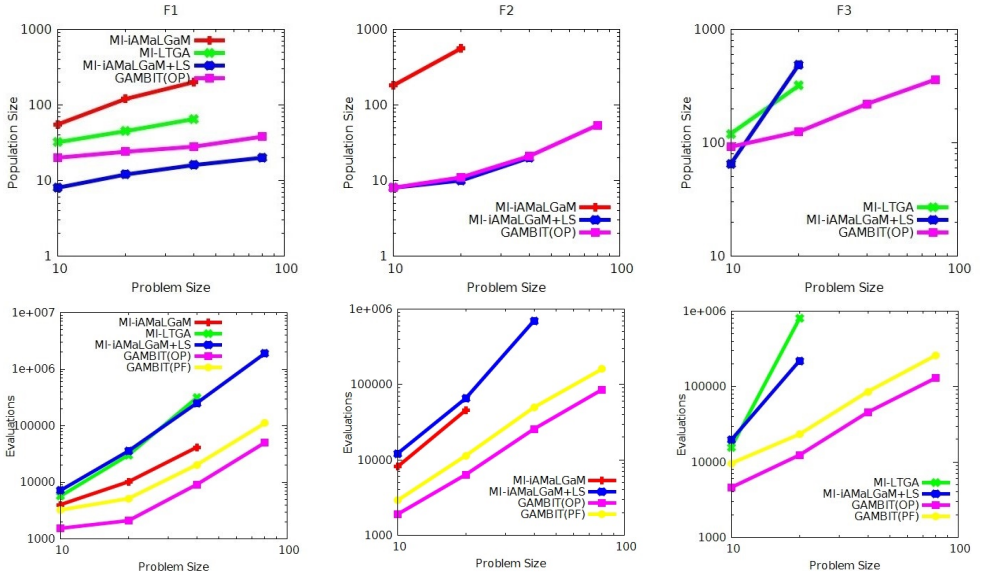


Figure 5.11: *Model-based EA variants performance for functions F_1, F_2, F_3.*

On the completely independent benchmark F_1 the algorithms perform relatively well, and manage to solve this simple benchmark for all tested problem sizes. MI-iAMaLGaM+LS requires smaller population sizes than other algorithms on F_1 and F_2. However, it performs significantly worse than the other algorithms in terms of function evaluations. MI-iAMaLGaM performs better than MI-iAMaLGaM+LS in terms of function evaluations on F_1 and F_2. However it is not able to consistently solve $F3$ - a problem where strong discrete dependencies are present. MI-LTGA can only solve small instances, and becomes very expensive as the problem size increases. GAMBIT is the most consistent and most efficient approach. It is able to solve all problems and performs better than other algorithms in terms of the number of evaluations needed.

Results of the experiments make it clear that single model-based approaches for mixed-integer problem optimization, while intuitive and straightforward, are not sufficient when different variable types are present. Discrete and continuous model-based EAs alone that we tested do not extend well into the mixed-integer domain. The results indicate that both of the algorithms are able to efficiently deal with the variables from the other domain only in a limited capacity. Once variable dependencies are introduced for the domain that the algorithm at hand does not build its

models for, originally, the respective algorithms become inefficient. By comparison, the additional computational mechanisms introduced in GAMBIT thus really appear to have substantial added value.

5.3.3 Inter-Variable Dependencies

In this section we focus on examining the added value of the inter-variable learning and sampling mechanism of GAMBIT. We consider problems with and without inter-domain dependencies and use different mixed-learning approaches in GAMBIT, including not considering mixed inter-variable dependencies at all. We also introduce a set of problems with constraints, where the level and types of dependencies are not explicitly known.

GAMBIT Variants

In addition to the full fledged GAMBIT variant we propose, we consider two variants that generate mixed subsets in similar fashion to GAMBIT. However, instead of the nearest-neighbor method, we utilize a much simpler mutual information estimation: fully discrete or fully continuous. For each of those variants, two ways of constructing the FOS structure is considered.

A fully discrete configuration refers to using the discrete mutual information calculation for all variables. This is accomplished with a crude discretization that calculates the average value of a variable in the current population, and assigns a zero value for all values below, and a one value for all values above the average. With these temporarily discretized variables, mutual information can be calculated in the same way as with LTGA, but now for all variables. In the fully continuous configuration, we do the converse. It is again possible to calculate mutual information between all variables if the discrete variables are treated as real-valued and using the definition of mutual information assuming a parametric distribution over the variables such as the normal distribution. For the purposes of this calculation, the continuous variables are normalized to a [0;1] range, and binary variables are treated as real values.

We consider two variants of generating the FOS, resulting in four algorithms. In the first variant we create a linkage tree with hierarchical clustering, but consider all problem variables. A tree obtained in this fashion contains $2(l_c + l_d)$-1 subsets, and can include mixed subsets. Depending if the mutual information was estimated entirely in the continuous or discrete domain, we refer to this approach as a fully continuous based FOS ("G-Full Cont.") or fully discrete based FOS ("G-Full Disc.") respectively. The second variant retains existing mechanisms for generating strictly discrete and strictly continuous subsets, and appends it with mixed subsets, which are built using an additional linkage tree. This approach allows us to retain all specific intra-domain dependencies, and join them with specifically learned inter-domain dependencies. In order to generate the strictly mixed-subset linkage tree, the mutual information between variables of the same type is set to $-\infty$, ensuring that in such a tree, all non-leaf nodes are guaranteed to contain both types of variables in each subset. The resulting mixed subsets are added to the FOS. Based on this FOS

representation we consider GAMBIT with the appended mixed subsets based on continuous estimation ("G-Joined Cont.") and discrete estimation ("G-Joined Disc.").

Constrained Problems and Constraint Handling

In addition to the benchmark set of unconstrained problems we also test GAMBIT with a selection of constrained problems. Constrained Problems (F_7 through F_{12}) vary in problem size, as described in Table 5.2. The other benchmarks (F_1 through F_4) were fixed to a problem size of 20 (the largest size for which we could test all the MINLP solvers in the GAMS framework), where half of the problem variables are continuous and the other half are discrete.

MI problems are often accompanied by constraints which limit the feasibility of the search space. In a black-box setting, we cannot make any assumptions about the number or types of constraints present in a given problem. The only feedback we may hope for is an indicator of how strongly the current problem constraints are violated. For such conditions, we consider the use of the Dynamic Penalty Function Method ([74],[76]).

The Dynamic Penalty Method worsens the value of the objective function if the solution is infeasible. This penalty factor is proportional to the constraint violation value as well to the number of generations which have already passed. This means that as time passes, the penalty factors of infeasible solutions becomes stronger. Because of these features, the penalty method initially allows for exploration of promising but not feasible regions. As time progresses however, feasible regions become more important. In our implementation the penalty function value is the square of the total constraint violation value multiplied by the number of generations which have already passed.

Results

The success criterion is finding a solution within 10^{-10} of the known global optimum at least 29 out of 30 times. A run is considered unsuccessful if the optimum is not found after 10 million evaluations or execution time exceeds an hour. A successful run for any tested algorithm was usually completed within a few minutes. The reported results for the unconstrained problems consist of the same number of discrete and continuous variables ($l_c = l_d = l/2$).

In Figure 5.12 "G-joined NN" refers to our fully-fledged proposed GAMBIT variant where the nearest-neighbor mutual information estimation is used to calculate mixed subsets which are added into the FOS. The "NoMixLearning" refers to a variant of GAMBIT without the learning mechanism for the mixed space. The remaining variants correspond to the four approaches detailed in the previous section.

Figure 5.12 shows that GAMBIT does not benefit from any variant of the mixed-dependency processing mechanism on the DT5Ellipse (F_4) problem, where no inter-variable dependencies exist, however it does on the OnemaxSphere (F_1) problem. This can be attributed to the larger number of subsets that exist per generation when the mixed dependency learning mechanisms are used. Because the subsets are correct, more subsets means more selection pressure used to get better solutions faster.

Table 5.2: *Specifications of the MINLP selected constrained benchmark problems, and initialization ranges for problem variables.*

Name	Function	Constraints	Range
F_7	$2r_1 + d_1$	$-r_1^2 - d_1^2 \leq -1.5, r_1 + d_1 \leq 1.6$	$r_1 \in [0, 1.6]$ $d_1 \in \{0, 1\}$
F_8	$2r_1 + 3r_2 + 1.5d_1 + 2d_2 - 0.5d_3$	$r_1^2 + d_1 = 1.25, r_2^{1.5} + 1.5d_2 = 3$ $r_1 + d_1 \leq 1.6, 1.333r_2 + d_2 \leq 3$ $-d_1 - d_2 + d_3 \leq 0$	$r_{1,2} \in [0, 10]$ $d_{1,2,3} \in \{0, 1\}$
F_9	$0.8 + 5(r_1 - 0.5)^2 - 0.7d_1$	$-exp(r_1 - 0.2) - r_2 \leq 0$ $r_2 + 1.1d_1 \leq -1, r_1 - 1.2d_1 \leq 0$	$r_1 \in [0.2, 1]$ $r_2 \in [0.22554, -1]$ $d_1 \in \{0, 1\}$
F_{10}	$6 + (r_1 - 1)^2 + (r_2 - 2)^2$ $+(r_3 - 3)^2 - d_1 - 3d_2 - d_3$ $-0.693147180559945d_4$	$r_1 + r_2 + r_3 + d_1 + d_2 + d_3 \leq 5$ $r_1^2 + r_2^2 + r_3^2 + d_3 \leq 5.5$ $r_1 + d_1 \leq 1.2, r_2 + d_2 \leq 1.8$ $r_3 + d_3 \leq 2.5, r_1 + d_4 \leq 1.2$ $r_2^2 + d_2 \leq 1.64, r_3^2 + d_3 \leq 4.25$ $r_3^2 + d_2 \leq 4.64$	$r_{1,2,3} \in [0, 10]$ $d_{1,2,3,4} \in \{0, 1\}$
F_{11}	$-5r_1 + 3r_2$	$8r_1 - 2r_1^{0.5}r_2 + 11r_2 + 2r_2^2 - 2r_2^{0.5} \leq 39$ $r_1 - r_2 \leq 3, 3r_1 + 2r_2 \leq 24$ $r_2 - d_1 - 2d_2 - 4d_3 = 1, d_2 + d_3 \leq 1$	$r_1 \in [1, 10]$ $r_1 \in [1, 6]$ $d_{1,2,3} \in \{0, 1\}$
F_{12}	$(r_4 - 1)^2 + (r_5 - 2)^2 + (r_6 - 1)^2$ $-log(1 + r_7) + (r_1 - 1)^2$ $+(r_2 - 2)^2 + (r_3 - 3)^2$	$r_1 + r_2 + r_3 + d_1 + d_2 + d_3 \leq 5$ $r_6^2 + r_1^2 + r_2^2 + r_3^2 \leq 5.5, r_1 + d_1 \leq 1.2$ $r_2 + d_2 \leq 1.8, r_3 + d_3 \leq 2.5, r_1 + d_4 \leq 1.2$ $r_5^2 + r_2^2 \leq 1.64, r_6^2 + r_3^2 \leq 4.25$ $r_5^2 + r_3^2 \leq 4.64$ $r_4 - d_1 = 0, r_5 - d_2 = 0$ $r_6 - d_3 = 0, r_7 - d_4 = 0$	$r_{1,2,3} \in [1, 10]$ $r_{4,5,6,7} \in [0, 1]$ $d_{1,2,3,4} \in \{0, 1\}$

Since all problem variables are independent, modeling dependencies over generations is not necessary.

On problems that contain inter-domain dependencies (F_5 and F_6) the advantages of the newly introduced mechanism become apparent. All variants of GAMBIT with the mixed subset processing mechanisms outperform the configuration without any mixed structure learning. The approach that uses discrete mutual information estimation performs worse than the remaining alternatives. This is likely due to the crude one bit discretization used in these approaches. The GAMBIT approach is the most efficient variant. This suggests that calculating mutual information in exclusively discrete or continuous domain results in a less accurate estimation of mixed dependencies than with the nearest-neighbor approach.

Furthermore, the G-Full Disc. and G-Full Cont. approaches fail to solve F_{12}. This suggests that relying on a single tree based FOS may be insufficient as intra-domain dependency information may be lost. Our observations are further confirmed on the constrained problems and summarized in Table 5.3. Dependency processing is less useful on problems with few variables. This can be seen in functions F_7 and F_{11}, where learning the mixed subsets is not advantageous. All the alternatives which attempt to estimate mixed dependencies are more successful than the ”NoMixedLearning” variant on most problems. The variant used with GAMBIT outperforms all alternatives on F_8, F_9, F_{11} and F_{12}, and therefore justifies our design choices.

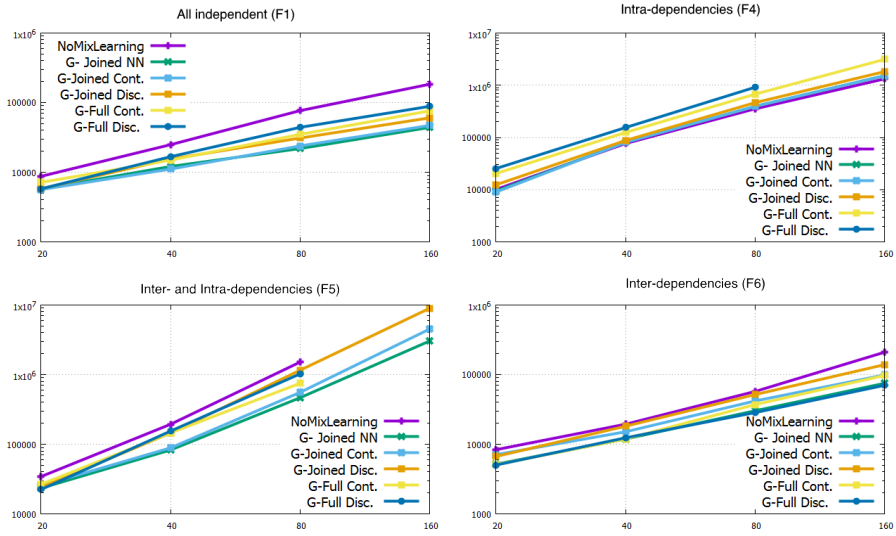


Figure 5.12: *Average number of evaluations required by GAMBIT variants on unconstrained benchmarks, over 30 runs.*

Table 5.3: *Performance of GAMBIT vs alternatives to learning mixed dependencies on constrained problems (average number of evaluations required, over 30 runs).*

	GAMBIT Variant					
Problem	**No Mixed Learning**	**G-Joined NN**	**G-Joined Cont.**	**G-Joined Disc.**	**G-Full Cont.**	**G-Full Disc.**
F7	**1056**	4096.7	4521	5836.3	4939	4015.8
F8	50431.5	**30868.3**	35976	35724.6	41953.2	62347.2
F9	3404.8	2909	2911	2819.1	3388.4	**2093.9**
F10	114887.2	**54645.3**	57855	75241.6	58098.1	119347.4
F11	6271.6	**4641.7**	**4693.3**	6709.5	**4771.6**	**4892.7**
F12	720494	**418224**	558219	6311296	x	x

5.3.4 Alternative Algorithms

While the mixed-integer optimization domain has been relatively less explored in the field of EAs than the individual discrete and continuous domains, certainly several alternative algorithms have been developed for the mixed-integer domain. In this section we briefly summarize several of such algorithms, which we use to compare and contrast results of GAMBIT with. Various state-of-the-art industrial mixed-integer optimization software tools exist, that are efficient and successful on a wide range of constrained and unconstrained problems. However, when faced with challenging non-convex problems, they may not perform as well.

Most of commercially available algorithms are known for their ability to efficiently solve problems by exploiting the structure of objective and constraint functions directly, i.e. taking a white-box approach. We wish to examine if these alternative approaches are able to exploit the structure of problems where dependencies between problem variables are a consequence of the objective function, or constraints.

We test the performance of a selection of Mixed Integer Nonlinear Programming (MINLP) solvers from the GAMS framework [22]. Additionally, we compare our results with the Mixed-Integer Evolution Strategy (MIES) ([57]).

Mixed Integer Evolution Strategy

The Mixed Integer Evolution Strategy, or MIES, is an evolution strategy approach to handling the mixed-integer domain in the black box setting. MIES extends the well-known $(\mu + \lambda)$ evolution strategy (ES) for continuous problems to mixed-integer search spaces. MIES generates solutions with the following procedure. The initial population is randomly generated. Two random solutions from the population $P(t)$ act as parent solutions. A recombination operator is applied to the parent solutions, followed by a mutation operator. These operators are defined differently for continuous, nominal discrete and integer problem variables and feature self-adaptive step-sizes and/or mutation probabilities. This procedure repeats until λ offspring solutions are created. The best μ solutions from the union of the μ parent solutions and the λ offspring are carried over into the new population $P(t+1)$ [57].

GAMS

General Algebraic Modeling System (GAMS) is a framework which gathers and allows access to a selection of commercial and open-source state-of-the-art mixed-integer solvers [22]. We consider the following GAMS algorithms that are designed for the mixed-integer non-linear programming (MINLP) problems:

- **ALPHAECP**
  The ECP method is an extension of Kelley's cutting plane method, which was originally given for convex NLP problems. ALPHAECP is an MINLP solver based on an extension of Kelly's cutting plane method (ECP) , originally developed for convex non-linear programming problems [92].

- **BONMIN**
  BONMIN, or the Basic Open-source Nonlinear Mixed Integer programming algorithm makes use of branch-and-bound, branch-and-cut, and outer approximation methods [9].

- **COUENNE**
  This Convex Over and Under Envelopes for Nonlinear Estimation algorithm [6], uses spatial branch-and-bound techniques. It can be viewed as an extension of BONMIN, which adds new functionality for computing linear outer approximations for non-convex problems and methods for bound tightening and branching on nonlinear variables.

- **DICOPT**
  DIscrete and Continuous OPTimizer is a mixed-integer optimization algorithm based on the extensions of the outer-approximation algorithm for the equality relaxation strategy [41].

- **KNITRO**
  A software package which makes use of interior-point and active-set methods for solving MINLP problems [25].
- **LINDO and LINDOGLOBAL**
  Algorithms adapting branch-and-cut methods [59].
- **LOCALSOLVER**
  This algorithm is based on a hybrid neighborhood search approach [8].
- **SBB**
  Algorithm based on the Standard Branch and Bound approach [24].
- **SCIP**
  Solving Constraint Integer Programs is based on the branch-cut-and-price algorithm [1] [35], and is considered one of the fastest non-commercial solvers for mixed-integer non-linear problems.

Results

Table 5.4 summarizes the results of testing the algorithms mentioned above on all considered benchmark problems. Almost without exception all of the algorithms have no trouble solving any of the constrained problems. The objective function definition of this selection of constrained benchmarks problems is relatively simple. Various equality and inequality constraints contribute to the difficulty of these problems. The algorithms in the GAMS framework are very well equipped to deal with such problems, as they very efficiently exploit many different constraint handling mechanisms. MIES, using the global competitive ranking introduced by [74] as a means of constraint handling, is also capable of solving $F_7 - F_{12}$ as explained in more detail by [57]. GAMBIT using a penalty method approach also solves the provided constrained benchmarks.

The performance of the tested algorithms changes however, when the objective function definition becomes more challenging, even in the absence of constraints. Benchmarks $F1$, $F2$ and $F3$ are solved by most algorithms tested. The MIES version provided by [57] does not solve $F_2 - F_5$. MIES's inability to solve F_2 can be remedied by the inclusion of covariance information during the generation of continuous variables. However, MIES cannot capture and exploit the Deceptive Trap function structure, which prevents it from solving $F_3 - F_5$.

When strong dependencies are introduced into the problem, the success rate of this set of algorithms goes down drastically. The $F4$ problem, which contains discrete and continuous dependencies localized to the respective domains is unsolvable for six out of the eleven tested algorithms. When cross-domain dependencies as well as discrete and continuous dependencies are featured, as in F_5, only two algorithms succeed.

In contrast, GAMBIT is capable of solving all of the problems in this benchmark set. A commercial algorithm ALPHAECP is the only other algorithm tested by us which also solved all the tested problems successfully. The runtimes of both algorithms were comparable. However, commercial algorithms strongly exploit the

Table 5.4: *Ability of MINLP algorithms to solve benchmarks with different landscape features.*

Function:	Unconstrained						Constrained					
Solver	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}	F_{11}	F_{12}
ALPHAECP	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BONMIN	✓	✓	**no**	**no**	**no**	**no**	✓	✓	✓	✓	✓	✓
COUENNE	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	**no**
DICOPT	✓	✓	**no**	**no**	**no**	✓	✓	✓	✓	✓	✓	✓
KNITRO	✓	✓	✓	**no**	**no**	✓	✓	✓	✓	✓	✓	✓
LINDO	✓	✓	✓	✓	**no**	✓	✓	✓	✓	✓	✓	✓
LINDOGLOBAL	✓	✓	✓	✓	**no**	✓	✓	✓	✓	✓	✓	✓
LOCALSOLVER	✓	✓	✓	**no**	**no**	✓	✓	✓	✓	✓	✓	✓
SBB	✓	✓	✓	**no**	**no**	✓	✓	✓	✓	✓	✓	✓
SCIP	✓	✓	✓	✓	**no**	✓	✓	✓	✓	✓	✓	✓
MIES	✓	**no**	**no**	**no**	**no**	✓	✓	✓	✓	✓	✓	✓
GAMBIT	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

structure of constraints as formalizable like in Table 5.2. GAMBIT does not use complex constraint-handling techniques. Instead, it treats the constrained space as a black box and utilizes the penalty function method and the clustering mechanism to solve constrained problems. Arguably, this makes GAMBIT more flexible when it comes to arbitrarily formulated types of constraints of which the formulation may not even be known.

5.4 Discussion

By design, GAMBIT aims to explore and exploit problem structure of multi-modal or otherwise challenging objective function landscapes which may contain inter- and intra-variable dependencies. Moreover, the design of GAMBIT was done with a black-box setting in mind. The problem types which GAMBIT aims to optimize differ from the problems usually tackled with the MINLP solvers in the GAMS framework. Such solvers are exceptionally equipped to optimize highly constrained problems, usually with relatively simple objective functions. They explicitly exploit linear constraints and do not treat the constrained space as a black box. This is demonstrated on problems F_7-F_{12}, where almost all GAMS solvers succeed on every constrained problem. The performance of GAMBIT on these functions is meant to illustrate GAMBITs ability to handle some constrained spaces despite treating the constraints as a black box. However, highly constrained MINLP problems solvable by the GAMS solvers remain out of reach for GAMBIT. Conversely, the results provided in Table 5.4 on functions F_1-F_6 show that while GAMBIT succeeds, many state-of-the-art algorithms struggle to solve even low dimensional problems when strong dependencies are present in the objective function between the discrete and continuous variables. GAMBIT does not assume anything about the objective function, the constraints, or about their structure. As such GAMBIT targets different types of

problems under different assumptions making a direct comparison to GAMS solvers not straightforward. This is not to say that GAMBIT can not be specialized to exploit known problem characteristics such as linear constraints either directly or by hybridization with other solvers, resulting in potentially vastly improved performance in such cases.

5.5 Summary

Mixed-Integer problems introduce many optimization challenges which do not arise in purely real-valued or discrete optimization problems. With the use of carefully designed benchmarks, we were able to showcase some of such challenges. Through consideration of a range of approaches we were able to narrow down a promising approach to tackle mixed-integer problems with model-based EA components. This approach, which we named the Genetic Algorithm for Model-Based mixed-Integer opTimization, GAMBIT, combines the structure learning abilities of LTGA and iAMaLGaM from the discrete and continuous domains respectively for use in the MI domain.

We observed that obtaining a proper balance in exploration of model information for different types of variables, varying variable ratios and additional overhead or fitness contribution scaling are some of the important issues which should be taken into account when solving mixed-integer problems. By extracting key features from EAs originally designed for either discrete or continuous domains, and carefully integrating their different models, we were able to develop a new EA and solve mixed integer benchmarks with varying degrees of variable dependencies. The resulting algorithm GAMBIT achieved polynomial scale-up behavior on the tested benchmarks. We believe that these results provide a good foundation and motivation for further work on solving mixed-integer optimization problems with model-based EAs.

Practicality is one of most important factors in real-world optimization. Many algorithms produce good results only when correctly parameterized, which can be problematic especially in black-box settings. Determining an optimal, or even a good population size parameter for an EA is not trivial. To address this, a parameter-free scheme was added to GAMBIT. Even though removing the need to specify the population size and number of clusters parameters from GAMBIT creates a potential need for additional evaluations, the scalability of the algorithm is only a small constant factor larger than using parameters which are empirically optimized beforehand. Optimizing such settings beforehand often limits the algorithm to one type of problem only, as such settings are often not transferable to other problems, further motivating a parameter-free approach.

The performance of a collection of state-of-the-art MI solvers was compared with GAMBIT. Our results showed that these solvers are very efficient on problems with simpler objective functions, even in the presence of constraints. However, when faced with non-convex objective landscapes, especially ones that contain strong intra-domain or cross-domain dependencies, most of the alternative approaches we considered cannot solve even low dimensional cases. GAMBIT performs much more efficiently on these benchmarks. Since real-world problems may exhibit difficult

landscapes with complex variable dependencies, and may be complex to an extent that a black-box approach is mandatory, our observations suggest that our model-based EA approach can be successful in the mixed-integer domain, also for real-world problems.

6 Multi-Objective GAMBIT

K. L. Sadowski, D. Thierens, and P.A.N. Bosman. Optimization of Multi-Objective Mixed-Integer Problems with a Model-Based Evolutionary Algorithm in a Black-Box Setting. *[Submitted for review (journal article)]*

In this chapter we bring the strengths of the recently-proposed Genetic Algorithm for Model-Based mixed-Integer opTimization (GAMBIT) to the multi-objective (MO) domain, and determine whether the superior performance of GAMBIT is maintained. In our approach, we introduce various mechanisms designed specifically for MO optimization resulting in MO-GAMBIT. We compare performance, in terms of the number of evaluations used and runtime, with alternative techniques, particularly linear scalarization and a selection of alternative MO algorithms. To this end, we introduce a set of objective functions which vary in composition in terms of discrete and continuous variables, as well as in the strength of dependencies between variables. Our results show that MO-GAMBIT can substantially outperform other algorithms, thereby providing a promising new approach for multi-objective mixed-integer optimization in a black-box setting.

6.1 Introduction

One of the key reasons why black-box optimization is considered to be challenging is because even when a problem is decomposable, if strong dependencies exist between problem variables, the scalability of algorithms becomes potentially very bad if such dependencies are not properly handled. Evolutionary Algorithms (EAs) are of particular interest in this setting, as they provide mechanisms which allow the learning and processing dependencies from a population, during optimization. The field of model-based EAs directly confronts the issue of not knowing key dependencies a priori. In the areas of continuous as well as discrete optimization, model-based EAs have already shown great potential in solving challenging academic and real-world problems in a black-box setting (see [16], [18] for examples of performance of EAs used in this chapter on continuous and discrete well-known benchmark problems respectively). Previous research has shown that the learning and model-building ability of such EAs extends successfully into the MO domain, where algorithms such

as MO-iAMaLGaM [14] and MO-GOMEA [62] perform efficiently in the MO continuous and discrete domains respectively.

Mixed-integer optimization in a black-box setting can be considered even more challenging, in large part due to the unknown dependencies that may exist not only *between* variables of the same type (intra-variable dependencies), but also *across* the discrete and continuous domains (inter-variable dependencies). Learning and exploiting potential variable dependencies is a well-known approach in the discrete and continuous optimization fields of evolutionary computation [89]. Less research exists however on this topic when facing mixed-integer problem landscapes where discrete and continuous variables are present simultaneously. Furthermore, existing optimization techniques in this domain are often focused on single-objective optimization, and do not explicitly consider multiple objectives.

Real-world optimization problems however often do have multiple objectives. Such objectives are moreover often in direct conflict with each other. A key characteristic of multi-objective optimization problems is that there is no single optimal solution, but rather a set of optimal solution exists in which different solutions trade-off the fitness of different objectives. The ultimate goal of MO optimization is to find the set of all Pareto-optimal solutions.

This chapter aims to design scalable EAs for problems where the following characteristics are present simultaneously: multiple objectives, mixed-integer landscapes and the black-box setting. By doing so, we wish to establish foundations for EAs that are capable of optimization of academic and real-world problems which exhibit all the above mentioned characteristics.

In the previous chapter, the Genetic Algorithm for Model-Based mixed-Integer opTimization (GAMBIT) was introduced for single-objective optimization in the black-box setting, and was shown to perform efficiently on various problems, outperforming other approaches, including straightforward combinations of evolutionary algorithms for different variable types with different levels of inter- and intra-variable dependencies (see also [79]).

GAMBIT combines and builds upon the learning and model building abilities of two EAs - LTGA [87] and iAMaLGaM [15], respectively designed for discrete and continuous variables, by introducing novel mechanisms designed for balanced discrete, continuous and mixed model learning and sampling. GAMBIT has shown promise in solving black-box problems where various intra-dependencies, as well as inter-dependencies are present in a problem.

By addition of mechanisms designed specifically for the MO domain, we introduce MO-GAMBIT. These mechanisms, explained in detail in further sections, include utilizing an Elitist Archive to retain and exploit best encountered solutions as well as different selection and variation operators.

We introduce a set of problems designed to highlight different types of mixed-integer problems with regard to problem composition and different types of possible variable dependencies in order to vary the types of dependencies present in a problem. We compare results with different types of alternative approaches.

Specifically, we consider a novel way of using repetition of the single-objective (SO) version of the same algorithm to search, in a balanced manner, for different points on the Pareto front. It is generally believed that a dedicated MO version

of an EA is more efficient, but here we test this hypothesis explicitly. To this end, we compare MO-GAMBIT with a more-traditional approach to MO optimization through linear objective scalarization using the single objective GAMBIT, as well utilizing Chebyshev scalarization. Additionally we consider a selection of other algorithms designed specifically for MO optimization, including iAMaLGaM adapted to mixed-integer problem landscapes, as well as a dedicated black-box MO solver for the mixed-integer domain: The Nonlinear Optimization by Mesh Adaptive Direct Search algorithm (NOMAD) [56].

Our results show that MO-GAMBIT performs better on the MO problems than different variants of the single-objective GAMBIT using objective scalarization. MO-GAMBIT is also capable of outperforming alternative algorithms designed for the MO domain considered in this chapter, especially when strong variable dependencies are present. Based on the results we find that MO-GAMBIT is a promising algorithm for black-box mixed-integer multi-objective optimization.

The remainder of this chapter is organized as follows. Section 2 introduces MO-GAMBIT and outlines all MO-specific mechanisms used. Alternative algorithms considered in this chapter as well the reasoning behind and specification of problems tested are detailed in Section 3. Section 4 summarizes the experimental results. Finally, Section 5 concludes our work.

6.2 Multi-Objective GAMBIT

MO-GAMBIT introduces new mechanisms to GAMBIT in order to handle multiple objectives. Some fundamental functionality of MO-GAMBIT however, is based on the existing, single-objective version of GAMBIT which was described in detail in Chapter 5 (also see [79]).

Evaluating the performance of an MO-EA is often not straightforward. With single-objective optimization, even in a black-box setting, any feasible solution can be evaluated and objectively ranked with respect to other feasible solutions thanks to the objective function value. EAs often rely on the ability to rank the solutions in a population, in order to guide the population towards exploring high-fitness regions of the search space. In the presence of multiple objectives ranking solutions becomes more difficult as solutions may exist, for which a value of one objective is high while simultaneously low for other objectives. This is especially common when faced with objectives which are conflicting. As a result, in vast majority of cases a single solution cannot be considered a comprehensive solution to a MO problem. Instead, a typical solution to a MO problem is represented as a set of so-called Pareto-optimal solutions, which form a front of optimal solutions. A theoretically optimal solution to a MO problem is a set of such non-dominated solutions which form a Pareto-front.

Specifically, MO optimization problems consist of m objectives $f_i(\boldsymbol{x}), i \in \{0, 1, \ldots, m-1\}$, that without loss of generality, must all be maximized. The objective value vector of a solution $\boldsymbol{x}$ is $\boldsymbol{f}(\boldsymbol{x}) = (f_0(\boldsymbol{x}), f_1(\boldsymbol{x}), .., f_{m-1}(\boldsymbol{x}))$. A solution $\boldsymbol{x}^0$ *dominates* a solution $\boldsymbol{x}^1$ (denoted $\boldsymbol{x}^0 \succ \boldsymbol{x}^1$) if and only if $f_i(\boldsymbol{x}^0) \geq f_i(\boldsymbol{x}^1), \forall i \in \{0, 1, \ldots, m-1\}$ and $\boldsymbol{f}(\boldsymbol{x}^0) \neq \boldsymbol{f}(\boldsymbol{x}^1)$. A solution $\boldsymbol{x}^0$ is Pareto optimal if and only if there does not exist a solution $\boldsymbol{x}^1$ such that $\boldsymbol{x}^1 \succ \boldsymbol{x}^0$. The Pareto set $\boldsymbol{\mathcal{P}_S}$ of the prob-

lem at hand is the set of all Pareto-optimal solutions. The Pareto front $\mathcal{P}_F$ is the set of the objective value vectors of all Pareto-optimal solutions. The goal of MO optimization is to find a set of non-dominated solutions whose objective value vectors constitute a good approximation of the Pareto front [28].

An estimation of a Pareto front is a set of solutions believed to lie on, or be very near to, the optimal front. Additionally, such solutions need to be spread out across the entire front, to accurately estimate it. In order to adequately compare the quality of solutions and the efficiently of the algorithms we consider the hypervolume metric [51]. Additionally, we compare the performance results of various algorithms in terms of the number of evaluation functions used as well as runtime. Details regarding these evaluation metrics are provided in Section 3.

In order to estimate the Pareto front, MO-GAMBIT uses a variety of mechanisms, which are detailed in this section.

6.2.1 Elitist Archive

In the literature on multi-objective EAs, elitism has been shown to be very important [12] [62]. For this reason, an elitist archive is used in MO-GAMBIT. Because real-valued variables are involved, we use a technique that adaptively changes the grid that governs the elitist archive so as to harbor a predefined maximum number of solutions, preventing occurrences of very similar solutions in the archive while promoting diversity. Specifically, every time a new solution is generated and evaluated, the elitist archive is updated. A new solution enters the elitist archive if it is not dominated by any already preserved solution (unless it lies too closely to an already existing solution, in accordance to the adaptively scaling precision). Any solutions which are dominated because of the addition of the new solution are deleted from the archive [61].

6.2.2 Selection and Variation

The selection and variation mechanisms are changed in MO-GAMBIT to account for the MO nature of the problems, and to make use of the elitist archive. Solutions are still clustered at every generation, however clustering is performed in the objective space. This allows MO-GAMBIT to specialize model-based optimization in different areas of the Pareto front. Variation proceeds by improving existing solutions, as is reminiscent of GOMEA, in case of discrete and mixed linkage sets. Solution parts are copied from donors and mixed into solutions, and changes are only accepted in case of a multi-objective improvement (i.e. Pareto-domination). Continuous variables are supplied from normal distributions estimated from MO rank-based selection [12] following procedures of the real-valued EDA known as iAMaLGaM [11] and its multi-objective counterpart iMAMaLGaM [10].

Upon sampling new solutions, the clusters are first populated with solutions from the elitist archive, using solutions that are closest to each cluster in the objective space, using a Euclidean distance metric. Up to a fraction of $\tau = 0.35$ solutions are copied this way. If not enough elitist archive solutions exist, the entire archive is used for each cluster and if further needed, solutions from the previous generations

are copied. This combination of solutions from the previous generation and the elitist archive is used to estimate and sample new solutions. Previously, this approach was found to have a positive effect on the speed of convergence to high-quality solutions [61, 10].

Population and Cluster Sizing

Determining parameters such as the population size or number of clusters can be crucial for the efficiency and effectiveness of EAs. Small population sizes or an insufficient number of clusters can lead to premature convergence. On the other hand, too many clusters or too large population sizes can result in a very large overhead. To avoid the need to specify these parameters, MO-GAMBIT adapts an interleaved multistart scheme, which dynamically introduces larger population sizes and more clusters. Specifically, the scheme runs differently-sized instances of GAMBIT in an inter-leaved fashion. For every γ generations of a certain instance of GAMBIT, one generation of an instance of GAMBIT with twice the population size and one more cluster is performed. By doing so, smaller population sizes perform more generational steps than larger ones. Moreover, in MO-GAMBIT the elitist archive is shared over all instances, allowing to transfer knowledge about high-quality solutions already obtained with smaller populations.

This scheme is illustrated in Figure 6.1. P_{min} represents an instance with a minimum population size. Each black rectangle represents execution of a single generation of the instance. The size of the rectangle represents the processing time / evaluations associated with the instance. Larger instances are created as they are needed, giving the illustrated pattern.

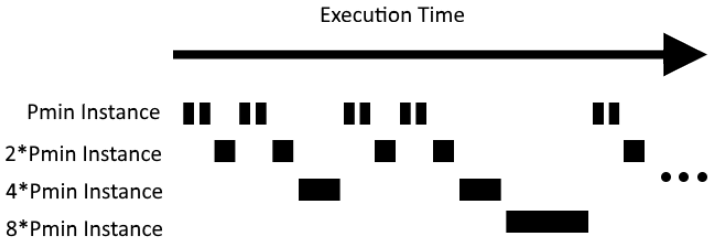


Figure 6.1: *Illustration of executing the interleaved multi-start scheme. P_{min} is an instance with a minimum population size.*

6.3 Setup of Experiments

We consider a set of algorithms of which the performance is compared to the newly introduced MO-GAMBIT. This section provides a summary and reasoning behind using these algorithms.

6.3.1 Algorithms

Repeated GAMBIT with Objective Scalarization

A relatively straightforward approach to solving MO problems is by transforming them into single objective problems and then use single-objective optimization approaches. To compare to such an approach, we introduce two variants of the single-objective GAMBIT (which from here on we will refer to as SO-GAMBIT) for the multi-objective domain.

In order to transform an MO problem into an SO problem a (weighted) scalarization of objectives can be used. We consider two common types of scalarization: one-norm and infinity-norm, resulting in two multi-objective variants of SO-GAMBIT respectively:

- SO-GAMBIT-One-Norm
- SO-GAMBIT-Infinity-Norm

In both of these variants weight vectors (w_1,w_2,...,w_m) are used, where $\sum w_i = 1$ and m is the number of objectives are constructed iteratively. For each weight vector SO-GAMBIT is run, resulting in a single point on the Pareto front.

The weighted objective function definitions, assuming minimization of each objective $f_i(x)$ and optimal value of 0, are as follows:

- One-Norm Scalarization: $f_{obj} = \sum w_i * f_i(x)$
- Infinity-Norm Scalarization: $f_{obj} = max_i|w_i * f_i(x)|$

Selecting the weight values is not straightforward and the need to determine their values can be considered a drawback of scalarization. A potentially intuitive, equidistant distribution of the weights may not be sufficient as the corresponding distribution on the Pareto front is not guaranteed to be evenly distributed. To address this, we consider a dynamic approach of determining the next set of weight distributions based on already discovered points of the Pareto front.

Specifically, at first the optimizer is ran m times, with each weight, in turn, set to 1. The center point in the m-dimensional simplex is then chosen as the next weight vector to evaluate, and the simplex is scattered into m new simplexes. This procedure is iterated, always choosing the largest simplex (in objective space) next.

MO-iAMaLGaM

MO-iAMaLGaM models a Gaussian distribution using Pareto-dominance based solution raking, clustering the objective space an generating new solutions via sampling the clustered distribution. Since this algorithm was originally designed to handle only continuous variables, adjustments are needed for the mixed-integer setting. Specifically, discrete variables are treated as continuous. Each binary variable is initialized randomly on a [0,1] range and treated as a continuous variable, but evaluated as either 0 or 1 depending on if the real value representation is above or below 0.5.

NOMAD

The Nonlinear Optimization by Mesh Adaptive Direct Search algorithm (NOMAD) is a mixed-integer solver. It includes MO handling mechanisms and is designed to solve both single and multi-objective problems. It has shown consistent and efficient results in both of these optimization areas. NOMAD implements the Mesh Adaptive Direct Search (MADS) algorithm [4], and is designed for black-box optimization under general nonlinear constraints. For more details on this algorithm, see [56].

6.3.2 Benchmark Problems

A mixed-integer problem can be defined as follows:

$$min\ F(\boldsymbol{x_d}, \boldsymbol{x_c})$$

$$\text{s.t. } \mathbf{h}(\boldsymbol{x_d}, \boldsymbol{x_c}) = 0,\ \ \mathbf{g}(\boldsymbol{x_d}, \boldsymbol{x_c}) \leq 0$$

Where

$$\boldsymbol{x_d} = d_0 ... d_{l_d - 1} \quad and \quad \boldsymbol{x_c} = c_0 ... c_{l_c - 1}$$

with $d_i \in \mathbb{D} \subseteq \mathbb{N}, c_i \in \mathbb{R}$ and $\boldsymbol{x_d}$, $\boldsymbol{x_c}$ are the groups of all discrete and continuous variables, respectively. F is the objective function. **h** and **g** are the sets of equality and inequality constraint functions respectively. If both of these sets are empty, the mixed-integer problem is considered to be unconstrained. In this work $\mathbb{D} = \{0, 1\}$.

Objective Functions

We consider a set of MO problems with objectives that have different types of variable dependencies and different numbers of objectives. To this end, the individual objectives that make up the MO problems used in this chapter are based on well-known discrete and continuous benchmark problems, see Table 6.1, and are combined in ways that capture different mixed-integer landscape features.

Table 6.1: *Continuous and Discrete functions used to define our unconstrained mixed benchmarks*

Discrete and Continuous Functions
$F_{Sphere}(x_c) = \sum_{i=0}^{l_c-1} c_i^2$
$F_{R.Ellip.}(x_c) = F_{Ellip.}(\mathbf{R} * x_c)$, where $F_{Ellip.}(x_c) = \sum_{i=0}^{l_c-1} 10^{6*i/(l_c-1)} * c_i^2$
$F_{Onemax}(x_d) = \sum_{i=0}^{l_d-1} d_i$
$F_{DT5}(x_d) = \sum_{i=0}^{l_d/k-1} f_{Trap-k}^{sub}(\sum_{j=ki}^{ki+k-1} d_j)$, where $f_{trap\ k}^{sub}(u) = \begin{cases} 0 & : if\ u = k \\ 1 - (k-1-u)/k & : otherwise \end{cases}$

These mixed-integer functions are presented in Table 6.2.

Table 6.2: *Unconstrained benchmark problems*

Definition	Landscape Feature
$F_1(\mathbf{x_d}, \mathbf{x_c}) = F_{Onemax}(\mathbf{x_d}) + F_{Sphere}(\mathbf{x_c})$	All Independent
$F_2(\mathbf{x_d}, \mathbf{x_c}) = F_{Zeromax}(\mathbf{x_d}) + F_{Sphere-1}(\mathbf{x_c})$	All Independent
$F_3(\mathbf{x_d}, \mathbf{x_c}) = F_{DT5_{inv}}(\mathbf{x_d}) + F_{R.Ellip.-1}(\mathbf{x_c})$	Continuous and Discrete Dependencies
$F_4(\mathbf{x_d}, \mathbf{x_c}) = F_{DT5}(\mathbf{x_d}) + F_{R.Ellip.}(\mathbf{x_c})$	Continuous and Discrete Dependencies

Multi-Objective Problems

In order to create multi-objective mixed-integer problems that explicitly feature different types of variable dependencies, the functions in Table 6.2 are used as individual objectives. Onemax and Sphere and well-know discrete and continuous optimization functions respectively, and are considered relatively simple as they consist of independent variables only. The Rotated Ellipsoid is a more difficult continuous function to optimize, as it contains very strong dependencies. A Deceptive Trap Function (DT) is a dependent discrete function. The DT5 function is a binary, non-overlapping, additively decomposable composition of the well-known deceptive trap function, with order $k = 5$, where problem length is $l_d = m * k$. Combining these functions in various ways allows us to explicitly control the type of variable interactions. We introduce the following MO mixed-integer problems: MO_1 - MO_3.

Each of the MO problems summarized in Table 6.3 represents different features of a problem landscape, based on the objective components they are made up of. In MO_1 all variables in each of the objectives are independent. In MO_2 variables of one of the objectives are independent, while the second objective consists of both, discrete and continuous, intra-variable dependencies. Finally, both objectives of MO_3 contain discrete and continuous intra-variable dependencies.

Table 6.3: *Multi-Objective Problems*

Name	Objectives
MO1	Obj1: $F_1(\mathbf{x_d}, \mathbf{x_c}) = F_{Onemax}(\mathbf{x_d}) + F_{Sphere}(\mathbf{x_c})$ Obj2: $F_2(\mathbf{x_d}, \mathbf{x_c}) = F_{Zeromax}(\mathbf{x_d}) + F_{Sphere-1}(\mathbf{x_c})$
MO2	Obj1:$F_1(\mathbf{x_d}, \mathbf{x_c}) = F_{Onemax}(\mathbf{x_d}) + F_{Sphere}(\mathbf{x_c})$ Obj2: $F_4(\mathbf{x_d}, \mathbf{x_c}) = F_{DT5}(\mathbf{x_d}) + F_{R.Ellip.}(\mathbf{x_c})$
MO3	Obj1: $F_3(\mathbf{x_d}, \mathbf{x_c}) = F_{DT5_{inv}}(\mathbf{x_d}) + F_{R.Ellip.-1}(\mathbf{x_c})$ Obj2: $F_4(\mathbf{x_d}, \mathbf{x_c}) = F_{DT5}(\mathbf{x_d}) + F_{R.Ellip.}(\mathbf{x_c})$

6.3.3 Evaluation Criteria

In order to measure the quality of solutions and the effectiveness of the algorithms tested, we consider the hyper-volume metric [51]. This metric, explained in detail in the following sections provides a much-used means of relative comparison between

different Pareto-front estimations. We consider the hyper-volume results against the number of function evaluations and computational runtime in our experimental setup.

Hyper-volume Metric

The hyper-volume is the volume of the objective space encompassed by an approximation front and a fixed Nadir point, i.e. a point that is always dominated by any point in the Pareto front. As all our benchmarks are normalized, the Nadir point always lies at a value of 1 for each objective. It ts worth noting that the actual value of the hyper-volume, which will be between 0 and 1, does not represent how good a given estimation is because the ideal hyper-volume value depends on the shape of the theoretical Pareto front and can vary for every problem. The hyper-volume value does however describe which estimation is better. Intuitively, if we compare two sets of Pareto front estimations with the same number of points, the one with the higher hyper-volume covers more volume, meaning the points are positioned closer to the optimal front, making the estimation a more accurate one.

Function evaluations and runtime

While the hyper-volume is the main indicator of the quality of the optimization results , we also look at other indicators of performance. Specifically, we examine the number of black-box evaluations needed by each algorithm to reach certain hyper-volume values. We also consider and contrast actual runtime.

6.3.4 Experimental Setup

Each experiment is repeated and averaged over 20 independent runs. Two different termination criteria are put in place: total number of evaluations and runtime. This is necessary due to substantial differences in the way that some of the tested algorithms behave. For example, MO-GAMBIT performs many evaluations in a short period of time, while NOMAD uses a lot more time and processing between evaluations. Due to such discrepancies, setting only either a time limit or an evaluation limit could bias the comparison by favoring one of the algorithms. Instead, a disjunction of both criteria is used as a termination condition. Specifically, the front estimation is terminated after half a million total evaluations or after 30 minutes of runtime per single run. All experiments were performed on a Intel(R) Dual Core(TM) i7-2760QM CPU@2.40GHz machine.

For each algorithm, default optimization settings were used. For the SO-GAMBIT algorithms, an evaluation cap is needed for every weight-combination optimization. Both SO-GAMBIT variants use $p = 200$ weight combinations, with each point alloted an equal number of evaluations totaling a half million.

6.4 Results

In this section we compare and contrast different algorithms. First, we consider the advantages and drawbacks of the MO mechanisms in MO-GAMBIT by comparing its performance to that of SO-GAMBIT variants as well as other algorithms designed for MO optimization: MO-iAMaLGaM and NOMAD as outlined in earlier sections.

Results in terms of the hypervolumes with respect to the number of black box evaluations as well as run times are summarized in Figure 6.3 for all problems. The results are summarized and discussed in the sections below.

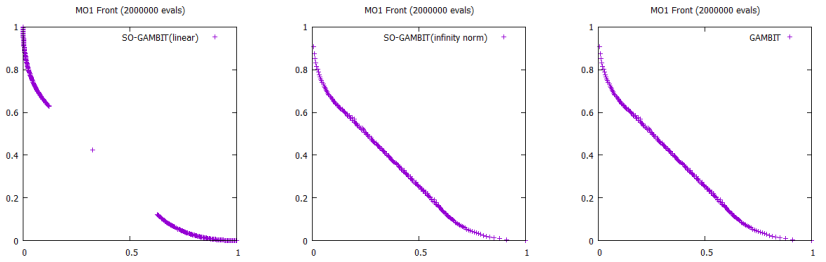


Figure 6.2: *Pareto-front estimation for MO1 for (a) SO-GAMBIT-One-Norm (b) SO-GAMBIT-Infinity-Norm (c) MO-GAMBIT*

6.4.1 Single vs. Multi-Objective GAMBIT

A weighted scalarization of objectives approach, such as both SO-GAMBIT variants, can be very powerful due its relative simplicity and the ability to use single-objective mechanisms directly to solve multi-objective problems. Results presented in Figure 6.3 show however that under certain circumstances such approach may be insufficient. This is already illustrated in the MO_1 problem. While the SO-GAMBIT-InfinityNorm approach is capable of reaching similar hypervolume levels as MO-GAMBIT, the One Norm variant is not. The reason for this is illustrated in Figure 6.2, where the approximation fronts generated by SO-GAMBIT and MO-GAMBIT are visualized.

Despite the relative simplicity of the components of MO_1, the resulting MO problem consists of both concave and convex regions on the Pareto front. As the One Norm approach is not capable of approaching convex parts of the front, the resulting estimation includes large gaps, which negatively affect the hypervolume metric results. When such parts of the front do not exist, exemplified in MO_2 results, the One Norm approach is capable of outperforming the Infinity Norm variant. Given the black-box setting, no assumptions can be made about the shape of the Pareto front, making the Infinity Norm a theoretically preferred option over the One Norm when solving MO mixed-integer problems. However, the infinity norm introduces discontinuities that are potentially far more difficult to deal with compared to using the one-norm variant, especially when using derivative-based local search operators.

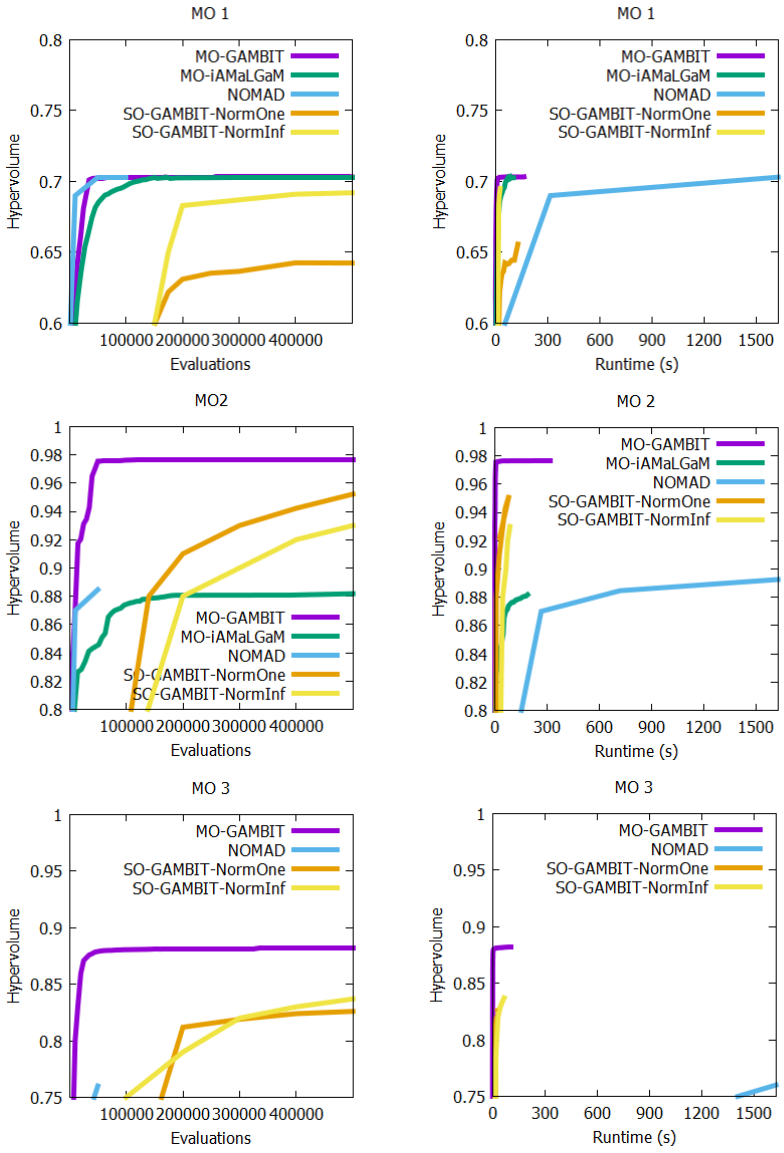


Figure 6.3: *Number of black-box evaluations and run-times vs. hyper-volume of the selected MO algorithms on problems MO_1-MO_3*

For SO-GAMBIT this appears to be of little concern, attesting to the competence of the GAMBIT design, in particular its variation operator.

MO-GAMBIT results show advantages of a dedicated multi-objective approach. On all three problems the overhead of MO mechanisms is outweighed by the resulting performance, as MO-GAMBIT outperforms both SO-GAMBIT variants. Given

enough time and evaluations, MO-GAMBIT and SO-GAMBIT-Infinity-Norm eventually converge to similar hyper-volume values. MO-GAMBIT is however much more efficient in doing so, in terms of both the number of evaluations needed and runtime. Moreover, SO-GAMBIT approaches require parametrization in terms of number of weighted combination considered and evaluations alloted for each. This is an additional difficulty given a black-box setting, where such parameters may be very difficult to predict. MO-GAMBIT does not need such parameterization.

6.4.2 MO Algorithm Performance

We further analyze the experimental results by considering other algorithms which, like MO-GAMBIT, are explicitly designed to handle multi-objective optimization, which are also illustrated in Figure 6.3 .

From the results it is clear that the MO-iAMaLGaM approach becomes very inefficient when facing problems with strong variable dependencies in the domain that they were not designed for. In terms of number of evaluations needed, NOMAD performs best on the MO1 problem, very quickly reaching high hyper-volume values. MO-GAMBIT requires more evaluations to reach the same values. However, in terms of runtime NOMAD is much slower than the remaining algorithms.

With strong variable dependencies present in MO_2 andMO_3, MO-GAMBIT outperforms the remaining algorithms in terms of both number of evaluations and runtime needed to reach high values of the hyper-volume.

6.5 Conclusions

Multi-objective mixed-integer optimization is often recognized as very challenging, especially under the assumption of black-box optimization. One characteristic which makes such problems difficult is the potential variable dependencies, resulting in complex search spaces and very poor behavior of optimization algorithms if these dependencies are not properly processed. In this chapter we introduced the evolutionary algorithm MO-GAMBIT which considers model-learning and processing features previously proven to result in efficient optimization of single-objective problems in combination with features specifically used in multi-objective optimization. In particular, MO-GAMBIT provides a novel, parameterless alternative to solving mixed-integer multi-objective problems.

Our results show that single-objective scalarization-based approaches are less efficient and require additional parametrization. Direct extensions of multi-objective algorithms targeted at one type of variable (i.e. continuous or discrete) will most likely also fall short when faced with problem landscapes with strong variable dependencies in the variable domain they were not designed for, as illustrated with the MO-iAMaLGaM.

Overall, MO-GAMBIT achieved favorable results compared to alternatives, including a well-known mixed-integer optimization algorithm for MO problems, NOMAD. Good results were achieved on problems with and without variable dependencies, regardless of whether the Pareto front is convex or concave. We therefore

believe that our results further motivate the use of MO-GAMBIT over all the alternatives considered in this chapter for mixed-integer MO optimization.

7 Application of GAMBIT to Brachytherapy Treatment Planning

This chapter is based on the following publication:

K. L. Sadowski, M. C. van der Meer, N. H. Luong, T. Alderliesten, D. Thierens, R. van der Laarse, Y. Niatsetski, A. Bel, and P. A. N. Bosman. Exploring Tradeoffs Between Target Coverage, Healthy Tissue Sparing, and the Placement of Catheters in HDR Brachytherapy for Prostate Cancer Using a Novel Multiobjective Model-based Mixed-integer Evolutionary Algorithm. In Proceedings of the Genetic and Evolutionary Computation Conference, GECCO '17, pages 1224—1231, 2017. ACM.

Brachytherapy is a form of radiotherapy whereby a radiation source is guided near tumors, using devices such as catheter implants. In the present clinical workflow, catheters are first placed inside or close to the tumor based on clinical expertise. Subsequently, software is used to design a plan for the delivery of radiation. Treatment planning is essentially a multi-objective optimization problem, where conflicting objectives represent radiation delivered to tumor cells and healthy cells. However, current clinical software collapses this information into a single-objective, constrained optimization problem. Moreover, catheter positioning is typically not included. As a consequence, it is hard to obtain insight into the true nature of the trade-offs between key planning objectives and the placement of catheters. Such insights are however crucial in understanding how better treatment plans may be constructed. To obtain such insights, we interface with real-world clinical software and derive potential catheter positions for real-world patients. Selecting and configuring catheters requires mixed-integer optimization. For this reason, we utilize the recently-proposed Genetic Algorithm for Model-Based mixed-Integer opTimization (GAMBIT) to tackle multi-objective optimization problems as presented in the previous chapter and apply it to the brachytherapy pre-planning problem. Our results indicate that clinically acceptable plans of high quality may be achievable with less catheters than typically used in current clinical practice.

7.1 Introduction

Prostate cancer is the most common type of cancer among men in the Netherlands, a diagnosis that over $10,000$ men are confronted with each year, and leading to about $2,500$ deaths per year [67]. Radiotherapy is a commonly applied cancer treatment in which ionizing radiation is used to target tumor cells. An important form of radiotherapy for prostate cancer is brachytherapy (BT), where the tumor is irradiated from inside the body by guiding a radiation source close to or inside the tumor. In this chapter, we focus on the case whereby catheters are placed in the prostate, for the radiation source to be moved through. Each catheter contains a number of different positions, called dwell positions, where the source can dwell for prescribed amounts of time, called dwell times. If the source dwells at a given dwell position, that position is called active; otherwise, it is called inactive. By varying the dwell times, different treatment plans are possible.

In determining a treatment plan, there are multiple aspects of importance. Firstly, as many tumor cells as possible should obtain a sufficiently high dose in order to maximize the probability of damaging or killing cancer cells. Secondly, the dose to nearby organs cannot be too high, to minimize the probability of complications. Most current clinical software for determining a treatment plan is based on single-objective optimization and manual tuning. However, because of the conflicting nature of these objectives, multi-objective optimization would be a more natural approach since trade-offs can be shown between the different objectives of interest, providing insight into how much may be gained in one objective at what cost to another. This way, a well-informed decision may be taken on what is the best treatment plan for the patient at hand.

Taking also the placement and number of catheters into account in the multi-objective optimization perspective is a natural fit, because the exact location of catheters governs much of what may be achieved in terms of plan quality. Moreover, to minimize the possibility of complications, one should use as few catheters as possible [90, 49, 31]. Currently, the placement of catheters is done mostly based on medical expertise. At the hospital involved in this study there are typically $16-18$ catheters used. Many placements are possible and for each placement many treatment plans are possible, comprising a vast search space. Even though expertise and literature exists on how to place catheters, there is little information on the real-world trade-offs of catheter placements. Our goal is to take a first step in this direction, obtaining such insights, learning about the complexity of the multi-objective optimization problem, and start to work toward algorithms that may really be used for patient-specific optimization in clinical practice. The novelty in our approach is bringing together several aspects that were not considered altogether before. Multi-objective approaches to treatment planning have been proposed, including catheter placement via the weighted-sum method in combination with single-objective optimization, but did not optimize directly on the evaluation criteria of a clinical evaluation protocol [50]. Related work that does optimize on these criteria exists, but is single-objective and does not consider catheter placement [30]. In contrast, we consider the use of real-world clinical software and patients combined with multi-objective optimization directly on criteria as considered in clinical evaluation proto-

cols.

From an algorithmic perspective, Evolutionary Algorithms (EAs) are among the state-of-the-art when it comes to solving multi-objective optimization problems [28]. For this reason, we are interested in using an EA to solve this problem. The problem includes both discrete (which catheters) and continuous (dwell times) variables. Many state-of-the-art (multi-objective) EAs focus on either discrete or continuous domains exclusively. Optimization where discrete and continuous variables are present simultaneously is explored relatively less and is referred to as mixed-integer optimization. A recently introduced Genetic Algorithm for Model-Based mixed-Integer opTimization (GAMBIT) has shown to be an effective approach to single-objective optimization in the mixed-integer domain [75], especially in the case of black-box optimization, meaning that no internal structure of the problem is assumed to be known in advance. This makes GAMBIT flexible and easily adaptable for our multi-objective approach of BT pre-planning optimization.

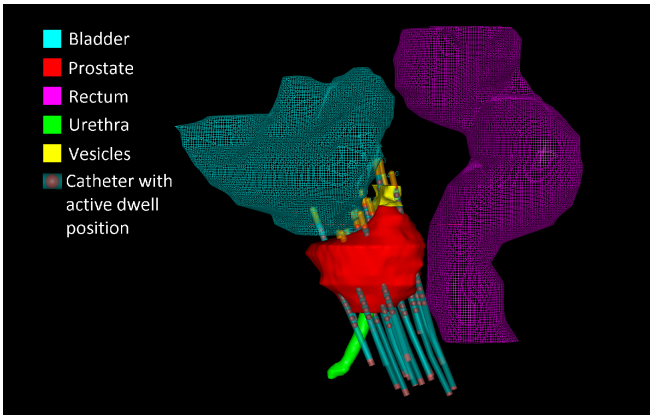


Figure 7.1: *Graphical representation of a patient's anatomy and implanted catheters.*

7.2 BT treatment planning

7.2.1 Clinical practice

In BT for prostate cancer, the treatment targets are the prostate and the seminal vesicles (see Fig. 1). The Organs At Risk (OARs), which should be radiated as little as possible, are the nearby organs, i.e., bladder, rectum, and urethra. Guided by live ultrasound images, a typical High-Dose-Rate (HDR) prostate BT treatment starts with implanting a number of catheters into the prostate through the patient's perineum skin (between scrotum and anus). The suitable number of catheters and their proper positions depend on the size of the prostate and its geometry in relation to other nearby organs (e.g., bladder and rectum). The inserted catheters need to be firmly fixed to prevent displacements. The patient is subsequently transferred to the imaging room, where computed tomography (CT) or Magnetic Resonance Imaging (MRI)

scans of the pelvic cavity are acquired. These scans are then used to finalize the treatment plan.

First, the catheters, the targets, and OARs are delineated on the CT/MRI scans in planning software. The doctors then specify which dwell positions in each catheter should be activated. For a given dwell time of a source at a dwell position, more radiation dose is delivered to nearby tissues than to faraway tissues. Therefore, to increase the probability of fully treating the target and sparing healthy tissues at the same time, dwell positions inside the target volumes are activated while the ones far away from the targets (e.g., at distances larger than 5mm) or too close to OARs are normally not considered. A trivial treatment plan, in which dwell times take on very large values, can kill all tumor cells but also causes necrosis (i.e., non-physiological cell death) to healthy tissues. On the other hand, another trivial treatment plan, where all dwell times are zero, can effectively spare healthy tissues from radiation risks but such a plan equals no treatment at all.

A specific setting of dwell times delivers a certain radiation dose distribution to the tissues surrounding the catheters. The dose distribution needs to be assessed by doctors to determine if such a setting can be clinically acceptable. A radiation dose is prescribed for the entire prostate, termed the *prescribed dose*. Tumor cells, which are more susceptible to radiation than healthy cells, should receive at least the prescribed dose to be effectively killed. Also, while healthy cells could recover from being radiated at the prescribed dose level, they should not be radiated too much; otherwise, healthy cells would be killed as well. However, it is impossible to compute the exact amount of radiation received by every cell since the number of cells is prohibitively numerous. Dose distributions, therefore, are often approximated by only computing the radiation dose at a certain number of so-called *dose calculation points*. Of key interest is how large the *cumulative* volume of an organ covered by a certain dose is. Such information, termed Dose-Volume-Histogram (DVH) indices, can be represented as V_x^o: the volume (expressed either relatively or absolutely) of organ o that received at least radiation dose level x. For example, international GEC/ESTRO recommendations [48] state that treatment plans should result in $V_{100}^{prostate} \geq 95\%$, i.e., the prostate volume that receives at least 100% of the prescribed dose should be $\geq 95\%$ of the total prostate volume. To prevent necrosis, it is required that $V_{200}^{prostate} \leq 20\%$, i.e., the prostate volume covered by 200% of the prescribed dose should $\leq 20\%$ of the total prostate volume. $V_{100}^{prostate}$, therefore, is a target coverage index while $V_{200}^{prostate}$ is an organ sparing index. Similarly, there exist other sparing DVH indices for other OARs. For example, it is recommended that $V_{110}^{urethra} \leq 0.1cc$, i.e., the urethra volume covered by 110% of the prescribed dose should be $\leq 0.1cc$. Finally, because the vesicles may also contain tumor cells, there is a constraint that $V_{80}^{vesicles} \geq 95\%$. A full description of the current clinical protocol in terms of DVH indices at the hospital involved in this study is presented in Table 7.1.

The values of DVH indices in the clinical protocol are consulted by the doctors when making treatment plans. Planning often starts with an initial plan proposed by BT treatment planning software. Note that it is difficult, however, to *directly* optimize DVH indices due to their discrete nature. For example, maximizing the $V_{100}^{prostate}$ index equals maximizing the *number* of dose calculation points inside the prostate

Prostate	Bladder	Rectum	Urethra	Vesicles
$V_{100} \geq 95\%$	$V_{86} \leq 1cc$	$V_{78} \leq 1cc$	$V_{110} \leq 0.1cc$	$V_{80} \geq 95\%$
$V_{150} \leq 50\%$	$V_{74} \leq 2cc$	$V_{74} \leq 2cc$		
$V_{200} \leq 20\%$				

Table 7.1: *BT treatment planning in clinical practice at the involved hospital: DVH index criteria.*

that receives at least the prescribed dose. To quickly achieve a result, planning software therefore often solves simplified optimization models of the problem with local search methods. It is thus not guaranteed that a treatment plan proposed by available software satisfies all DVH index criteria in the protocol or that the plan is the best possible one for the patient case at hand. Medical planners then need to manually adjust this proposed plan. They often first consider the DVH index that is violated the most compared to the recommended protocol, and try to find the locations of violation, then change the dwell times of nearby dwell positions to improve the index until satisfied. They then continue with a different DVH index that is now the most violated. Note that improving a DVH index might deteriorate other indices, including ones that have been previously worked on. The plan is adjusted in such an iterative manner until the doctors are satisfied.

How good the DVH indices can be, depends on the quality of the catheter implant and the geometry of the involved organs. Certain implants will not allow all DVH indices to satisfy the recommended protocol. For example, with too few or improperly placed catheters, it is difficult to obtain $V_{100}^{prostate} \geq 95\%$ while meeting constraints on indices of OARs. Similarly, if the rectum is too close to the prostate, $V_{78}^{rectum} \leq 1cc$ and $V_{100}^{prostate} \geq 95\%$ might not be achievable at the same time. In such situations, for each specific case, the doctors need to decide which indices are more important to be satisfied and which indices can be compromised. Final approved treatment plans, thus, might (slightly) violate some clinical protocol thresholds.

7.2.2 Multi-Objective Treatment Planning

BT treatment planning is intrinsically a multi-objective optimization problem. The DVH indices in Table 7.1 can be categorized into two groups: treatment target indices (i.e., $V_{100}^{prostate}$ and $V_{80}^{vesicles}$) and organ sparing indices. Candidate treatment plans that do not achieve the minimum requirements on the indices in the clinical protocol are less favorable than candidate plans that satisfy all protocol thresholds. For a treatment target index, the larger its value is (above the corresponding threshold), the better it is. For an organ sparing index, the lower its value is (below the corresponding threshold), the better it is. Optimizing treatment target indices results in increasing dwell times to make the target volumes covered by the treatment dose level as large as possible, where the best value is 100%. On the other hand, optimizing organ sparing indices results in decreasing dwell times to make the organ volumes covered by the radiation risk dose levels as small as possible, where the best value is $0\%/0cc$. The two groups of DVH indices conflict with each other such that a *utopian* treatment plan yielding 100% for all target coverage indices and $0\%/0cc$ for

all organ sparing indices does not exist. Instead, the optimum of this multi-objective problem is a set (i.e., *Pareto* set) of equally-good alternative plans which are optimal in the sense that improving one objective deteriorates the other objective, and vice versa.

There are nine different DVH index criteria. We argue, however, that the clinical protocol can be reformulated to a bi-objective optimization problem without losing key insight into trade-offs by combining the DVH indices of the same group (i.e., target coverage or organ sparing) into a representative objective of that group. To this end, we employ a risk-averse perspective, meaning that a treatment plan is evaluated to be only as good as the worst target coverage index and the worst organ sparing index. More specifically, each candidate plan has two objective values: the *Least Coverage Index* (LCI), which corresponds to the worst-scored DVH index in the target coverage group, and the *Least Safe Index* (LSI), which corresponds to the worst-scored DVH index value in the organ sparing group. Therefore, for a specific treatment plan, all indices in a group are at least as good as the representative value of the group. Such an approach has an analogy with the clinical practice in the iterative focus on improving the current worst index at each time as mentioned previously. Moreover, unlike the weighted-sum approach, where all criteria are added together, each DVH index in our problem modeling is still, in effect, considered separately.

The feasible search space is enlarged to contain solutions that violate the clinical protocol to some degree. There are two reasons for this. The first is that, in this way, the clinically-feasible solutions can be approached from both the clinically-feasible and the clinically-infeasible space, making the search in these regions more efficient. Secondly, we account for the fact that sometimes there exists no treatment plan satisfying all clinical protocols (due to the quality of the implants and/or the geometry of the involved organs, see Section 2.1), and a treatment plan that (slightly) violates the protocol must thus be accepted. To relax the clinical constraints, the new upper bounds for organ sparing indices are increased four times (e.g., $V_{86}^{bladder} \leq 1cc$ in the protocol becomes $V_{86}^{bladder} \leq 4cc$ in the optimization model). The new lower bounds for target coverage indices are also decreased accordingly as: $V_{100}^{prostate} \geq 80\%$ and $V_{80}^{vesicles} \geq 80\%$. The optimization constraints are presented in Table 7.2. Let V_x^{max} denote the clinical protocol threshold of a dose level x for an organ. To measure and normalize the distance of an organ sparing index value to its corresponding upper bound, we define $\delta(V_x) = 1 - \frac{V_x}{4 \times V_x^{max}}$. The larger the value of δ is, the better the corresponding DVH index value V_x of that organ is and how *safe* an organ is compared to the relaxed protocol threshold, with a value of 0.75 meaning that the clinical protocol constraint is met. The LSI objective value is defined as the minimum δ value among all organ sparing indices. Similarly, the LCI objective value of a treatment plan is defined as the minimum coverage value among all target coverage indices. For example, a treatment plan with $V_{100}^{prostate} = 97\%, V_{80}^{vesicles} = 96\%, V_{150}^{prostate} = 19\%, V_{200}^{prostate} = 7\%, V_{86}^{bladder} = 0.5cc, V_{74}^{bladder} = 1.3cc, V_{78}^{rectum} = 0.4cc, V_{74}^{rectum} = 1.2cc, V_{110}^{urethra} = 0.1cc$ would have the Least Coverage Index value as 0.96 (taking the value of $V_{80}^{vesicles}$) and the Least Safe Index value as 0.75 (corresponding with $V_{110}^{urethra}$).

In this chapter, we focus on the pre-planning phase, i.e., before the catheters are inserted. To generate data and problem input that corresponds to real-world clinical

Prostate	Bladder	Rectum	Urethra	Vesicles
$V_{100} \geq 80\%$	$V_{86} \leq 4$cc	$V_{78} \leq 4cc$	$V_{110} \leq 0.4cc$	$V_{80} \geq 80\%$
V_{150} free	$V_{74} \leq 8cc$	$V_{74} \leq 8cc$		
$V_{200} \leq 80\%$				

$$
\begin{aligned}
f_1 &= \min\{V_{100}^{prostate}, V_{80}^{vesicles}\} \\
f_2 &= \min\{\delta(V_{150}^{prostate}), \delta(V_{200}^{prostate}), \delta(V_{86}^{bladder}), \delta(V_{74}^{bladder}), \delta(V_{78}^{rectum}), \\
&\qquad \delta(V_{74}^{rectum}), \delta(V_{110}^{urethra})\} \\
f_3 &= \text{number of catheters used.}
\end{aligned}
$$

Table 7.2: *Multi-objective optimization model for BT treatment planning. f_1 and f_2 should be maximized while f_3 should be minimized. Because of the relaxation factor, $V_{150}^{\text{prostate}}$ has become unconstrained.*

practice, we are currently bound to real-world clinical software. Given a prostate BT patient, a number of possible catheter configurations can be determined, after which the clinical software provides information on these possible catheter configurations and their corresponding dwell positions. Hence, there is a restriction in the sense that the optimization can only use pre-determined catheter positions. In the software many catheters can be placed, but in a real patient it is desirable to have catheters not too close to each other to avoid complications. In order to have both a large enough search space for interesting first results and a set of configurations in which catheters are not too close to each other, we chose the set of possible configurations to consist of 30 catheters for each patient.

The optimization problem now contains d discrete and c continuous variables. The d discrete variables represent all the possible catheter placement positions. Each of these variables takes on a binary value, where 1 represents placing the corresponding catheter, and 0 not placing the catheter. Each of the continuous variables represents the dwell time corresponding to a dwell position in a catheter. Dwell times for dwell positions inside an inactive catheter can be ignored.

7.3 Model-based Multi-Objective Mixed-Integer Evolutionary Algorithm

In this section we summarize key concepts and terminology pertaining to mixed-integer and multi-objective optimization and provide an overview and summary of GAMBIT, and introduce the Multi-Objective GAMBIT (MO-GAMBIT).

7.3.1 Terminology

Mixed-integer optimization

A mixed-integer problem is defined as follows:

$$max\ f(x_d, x_c)$$

$$\text{s.t. } \mathbf{h}(x_d, x_c) = 0, \;\; \mathbf{g}(x_d, x_c) \leq 0$$

Here, x represents the solution

$$x = x_d x_c = d_0 ... d_{l_d - 1} \; c_0 ... c_{l_c - 1}$$

where $d_i \in \{0, 1\}, c_i \in \mathbb{R}$, and x_d, x_c are the groups of all discrete and real-valued variables, respectively. l_d and l_c represent the number of discrete and continuous variables. Moreover, f is the objective function, and **h** and **g** are the sets of equality and inequality constraint functions, respectively. If both sets are empty, the mixed-integer problem is said to be unconstrained.

Multi-Objective Optimization

A Multi-Objective (MO) optimization problem consists of m objectives $f_i(\boldsymbol{x}), i \in \{0, 1, \ldots, m-1\}$, that without loss of generality, must all be maximized. The objective value vector of a solution $\boldsymbol{x}$ is $\boldsymbol{f}(\boldsymbol{x}) = (f_0(\boldsymbol{x}), f_1(\boldsymbol{x}), \ldots, f_{m-1}(\boldsymbol{x}))$. A solution $\boldsymbol{x}^0$ *dominates* a solution $\boldsymbol{x}^1$ (denoted $\boldsymbol{x}^0 \succ \boldsymbol{x}^1$) if and only if $f_i(\boldsymbol{x}^0) \geq f_i(\boldsymbol{x}^1), \forall i \in \{0, 1, \ldots, m-1\}$ and $\boldsymbol{f}(\boldsymbol{x}^0) \neq \boldsymbol{f}(\boldsymbol{x}^1)$. A solution $\boldsymbol{x}^0$ is Pareto optimal if and only if there does not exist a solution $\boldsymbol{x}^1$ such that $\boldsymbol{x}^1 \succ \boldsymbol{x}^0$. The Pareto set $\boldsymbol{\mathcal{P}_S}$ of the problem at hand is the set of all Pareto-optimal solutions. The Pareto front $\boldsymbol{\mathcal{P}_F}$ is the set of the objective value vectors of all Pareto-optimal solutions. The goal of MO optimization is to find a set of non-dominated solutions whose objective value vectors constitute a good approximation of the Pareto front [28].

7.3.2 GAMBIT

GAMBIT is a recently introduced EA aimed at solving mixed-integer problems, especially in the black-box setting. Recall the detailed specification and design of GAMBIT is explained in detail in Chapter 5 of this thesis. Here, we give a brief outline of GAMBIT as it pertains to brachytherapy. GAMBIT is a parameter-free model-based EA capable of learning and exploiting different types of variable dependencies, through a clustering mechanism and an integrated dependency-models mechanism. The ability to learn such dependencies (i.e., which groups of variables have a joint synergistic effect on a solution's quality) allows for the creation of variable subsets which represent important building blocks of the problem. These building blocks can be used to exploit problem structure and generate new and better solutions more effectively. Such exploitation of problem structure based on dependencies is well-established for discrete and continuous variables, respectively. Mixed-integer optimization introduces new optimization challenges, however. The existence of discrete and continuous variables creates a potential for intra- and inter-variable dependencies that domain-specific model-based EAs are not equipped to handle. GAMBIT therefore introduces a new way to learn and process mixed inter-domain variable dependencies, alongside intra-domain dependencies (i.e., in individual domains) during optimization.

GAMBIT splits the population in each generation with the use of a clustering algorithm as detailed in Figure 7.2. Each of the sub-populations acquired is then

subject to variation through the integrated dependency models mechanism, summarized in pseudo-code in Figure 7.3. Specifically, a Family-Of-Subsets (FOS) model is learned at the beginning of every generation, which is used to describe dependencies and to create offspring solutions [88]. Essentially, a FOS is a set of linkage sets. Each linkage set contains the indices of the variables that are considered to be dependent. The linkage sets may overlap. The FOS consists of three types of subsets: discrete, continuous, and mixed. In its full black-box optimization configuration discrete subsets are generated using a so-called linkage tree, which is a specific type of FOS that was first used in the algorithm known as Linkage Tree Genetic Algorithm (LTGA), which is now considered to be an instance of the Gene-pool Optimal Mixing Evolutionary Algorithm (GOMEA) family [88]. The linkage tree is a FOS structure acquired by building from the bottom up (i.e., starting with singleton subsets of problem variables) a tree of subsets by means of a hierarchical clustering algorithm based on mutual information between pairs of variables. The linkage tree has $2l_d - 1$ discrete subsets representing important building blocks in the discrete domain. The same number of continuous subsets are added to the FOS, each containing every continuous problem variable. Additionally, $l_c + l_d$ mixed subsets are added to the FOS by building another linkage tree constrained to merge discrete and continuous variables using a mixed mutual information metric, described in detail in [75]. Such mixed subsets allow for the consideration of discrete and continuous variables together, resulting in the ability to exploit potential mixed variable dependencies. Each type of subset is processed with a corresponding mechanism type to generate new solutions. Once all subsets are processed, new solutions replace the previous population and the clustering process begins anew.

GAMBIT Overview

$\mathcal{P} \leftarrow$ GENERATEANDEVALRANDOMPOPULATION(n)
while $\neg$TERMINATIONCRITERIONSATISFIED **do**
  $\mathcal{C} \leftarrow$ DETERMINECLUSTERCENTERS$(\mathcal{P}, k)$
  for $j \in \{0, 1, \ldots, k-1\}$ **do**
    $\mathcal{P}_k \leftarrow$ GROWSUBPOPFROMCENTERS$(\mathcal{P}, \mathcal{C}_k)$
    $\mathcal{O}'_k \leftarrow$ INTEGRATEDMODELSMECHANISM$(\mathcal{P}_k)$
  $\mathcal{P} \leftarrow \mathcal{O}'_0 \cup \mathcal{O}'_1 ... \cup \mathcal{O}'_{k-1}$

Figure 7.2: *Pseudo-code overview of GAMBIT. $\mathcal{P}$ represents the population of size n. $\mathcal{C}$ contains k cluster (also called sub-population) centers. $\mathcal{P}_i$ represents the i-th sub-population and $\mathcal{O}'_i$ is the offspring generated from sub-population $\mathcal{P}_i$.*

7.3.3 Multi-Objective GAMBIT

In this chapter we utilize MO-GAMBIT that brings the black-box mixed-integer optimization capabilities of GAMBIT into the MO domain, as introduced in the previous chapter. The approach to handle MO problem landscapes with GAMBIT is to include mechanisms designated specifically to exploit the multi-objective nature of the problem landscape, such as an elitist archive and the ability to rank solutions in a multi-objective space. In this section, we outline and refresh the key components

Integrated Model Mechanism for a sub-population $\mathcal{P}_k$

```
INTEGRATEDMODELMECHANISM(P_k)
  F = LearnFOS(P_k)
  for i ∈ {0, 1, ..., (|F| − 1)} do
    if F^i is all continuous then
      S ← TRUNCATIONSELECTION(P_k, τ)
      UPDATECONTINUOUSMODEL(k, S)
      for j ∈ {0, 1, ..., n − 1} do
        ((P_k)_j)_{F^i} ← SAMPLECONTINUOUSMODEL(k, F^i)
        EVALUATEFITNESS((P_k)_j)
    if F^i is all discrete then
      for j ∈ {0, 1, ..., n − 1} do
        O ← (P_k)_j
        donor ← GETRANDOMSOLUTION(P_k)
        (O)_{F^i} ← (donor)_{F^i}
        EVALUATEFITNESS(O)
        if fitness(O) at least as good as fitness((P_k)_j) then
          ((P_k)_j)_{F^i} ← (O)_{F^i}
    if F^i is mixed then
      P_{k_sub}[0, ..., k − 1] ← GROUPINTOCLUSTERS(i, k, P_k)
      UPDATEMIXEDSUBMODELS(i, P_{k_sub})
      for j ∈ {0, 1, ..., n − 1} do
        SubID ← DETERMINESUBPOPULATION(j)
        O ← (P_k)_j
        (O)_{F^i} ← SAMPLESUBMODEL(i, P_{k_sub}[SubID])
        EVALUATEFITNESS(O)
        if fitness(O) at least as good as fitness((P_k)_j) then
          ((P_k)_j)_{F^i} ← (O_k)_{F^i}
  return P_k
```

Figure 7.3: *Pseudo-code for generating solutions with GAMBITs Integrated Models Mechanism.*

used in the design of MO-GAMBIT as introduced earlier in this thesis and outline its specific application to brachytherapy pre-treatment planning.

Elitist Archive

In previous work on multi-objective EAs, elitism has been shown to be very important [12]. For this reason, an elitist archive is added in MO-GAMBIT. Because real-valued variables are involved, we use a technique that adaptively changes the grid that governs the elitist archive so as to harbor a predefined maximum number of solutions, preventing occurrences of very similar solutions in the archive while promoting diversity. For more details, see [61].

Selection and Variation

The selection and variation mechanisms are changed in MO-GAMBIT to account for the MO nature of the problems, and to make use of the elitist archive. Solutions are still clustered at every generation, however clustering is performed in the objective space. This allows MO-GAMBIT to specialize model-based optimization in differ-

ent areas of the Pareto front. Variation proceeds by improving existing solutions, as is reminiscent of GOMEA, in case of discrete and mixed linkage sets. Solution parts are copied from donors and mixed into solutions, and changes are only accepted in case of a multi-objective improvement. Continuous variables are supplied from normal distributions estimated from a population-based MO rank-based selection [12] following procedures of the real-valued EDA known as iAMaLGaM [11] and its multi-objective counterpart iMAMaLGaM [10]. For an overview in pseudocode, see Figure 7.3.

Upon sampling new solutions, the clusters are populated with solutions from the previous generation as well as a fraction of $\tau = 0.35$ elitist archive solutions that are closest to the given cluster in the objective space, using a Euclidean distance metric. If not enough elitist archive solutions exist, the entire archive is used for each cluster. This combination of solutions from the previous generation and the elitist archive is used to estimate and sample new solutions. Previously, in MO versions of GOMEA and iAMaLGaM this was found to have a positive effect on convergence to high-quality solutions [61, 10].

Population and Cluster Sizing

Determining parameters such as the population size or number of clusters can be crucial for the effectiveness of EAs. Small population sizes or insufficient number of clusters can lead to premature convergence. Conversely, too many clusters or too large population sizes can result in a very large overhead. To avoid the need to specify these parameters, MO-GAMBIT adapts a interleaved multistart scheme, which dynamically introduces larger population sizes. Specifically, the scheme runs differently sized instances of GAMBIT in an inter-leaved fashion. For every $\gamma = 2$ generations of a certain instance of GAMBIT, one generation of an instance of GAMBIT with twice the population size and one more cluster is performed. By doing so, smaller population sizes perform more generational steps than larger ones. The scheme already used with GAMBIT is altered to make use of the elitist archive. With GAMBIT, all the differently sized instances are completely independent of each other. In MO-GAMBIT the elitist archive is shared over all instances, allowing to transfer knowledge about high-quality solutions already obtained with the smaller populations.

Problem-Specific Knowledge

A strictly black-box optimization algorithm may be a great starting point when considering new problems. In a real-world setting however, some problem-specific information can potentially improve performance. In our case consider that every discrete variable $d_0, d_1, ..., d_{l_d-1}$ is associated with an independent set of dwell positions. Instead of learning mixed-variable dependencies, which could result in a significant overhead, a pre-defined mixed-variable subset structure is defined that captures key dependencies. Specifically, each subset contains one discrete variable (representing the catheter) and the set of continuous variables that represent the dwell positions associated with this catheter. Moreover, because we do not expect strong dependencies to exist between continuous variables, we do not model them jointly

in one linkage set as per default in GAMBIT, but rather model them independently, i.e., univariately to reduce substantially the overhead of sampling high-dimensional Gaussian distributions.

Sliced 3-objective optimization

We minimize the number of catheters in one of the objectives. However, we slice this 3-dimensional problem into many 2-dimensional multi-objective problems. Little relation between configurations with different numbers of catheters exists, and switching a catheter on (i.e., changing a binary variable from 0 to 1) has huge impact, implying there exist big jumps in the search space across different numbers of catheters, making a direct 3-objective optimization approach extra difficult. A sliced approach allows an instance of GAMBIT to focus on configurations with the same number of catheters, while at the same time making it easier to parallelize the optimization across configurations with different number of catheters. We refer to our approach as a multi-layer approach, where in a given layer only combinations with the of number of catheters are allowed. Because solutions resulting in a different numbers of catheters can be created in variation, a simple random repair mechanism is used to activate or deactivate a required number of catheters.

7.4 Experiments

7.4.1 Setup

Clinical data from three recently treated patients is used for optimization. For each patient, there are 30 possible catheter locations, resulting in 30 MO-GAMBIT instances, each performing optimization on a 2-objective mixed-integer problem. The duration of optimization for each MO-GAMBIT instance is limited to 48 hours (on an AMD Opteron(tm) Processor 6386 SE and Intel(R) Xeon(R) CPU E5 2699 v4), with all 30 configurations executed in parallel. During the optimization 20,000 randomly chosen dose calculation points are used to compute the objective values for every function evaluation. To ensure sufficient accuracy in reporting final outcomes, the final set of solutions on the approximation front is re-evaluated with 100,000 dose calculation points, which is commonly considered to be sufficiently accurate.

7.4.2 Results

Results in Figure 7.4 illustrate the quality of solutions which can be acquired with different numbers of catheters. The results provide a clear illustration of the trade-offs between the objectives. A solution that satisfies clinical protocol lies in the area of the approximation front where LCI ≥ 0.95 while LSI ≥ 0.75. The clinically approved solutions for the considered patients all used 16 catheters. Figure 7.4 shows that it is possible to obtain approvable, high-quality plans using smaller numbers of catheters. Very little quality of solutions is lost when using a few less catheters. Only when many less catheters are used (less than 9 for the tested patients) the resulting

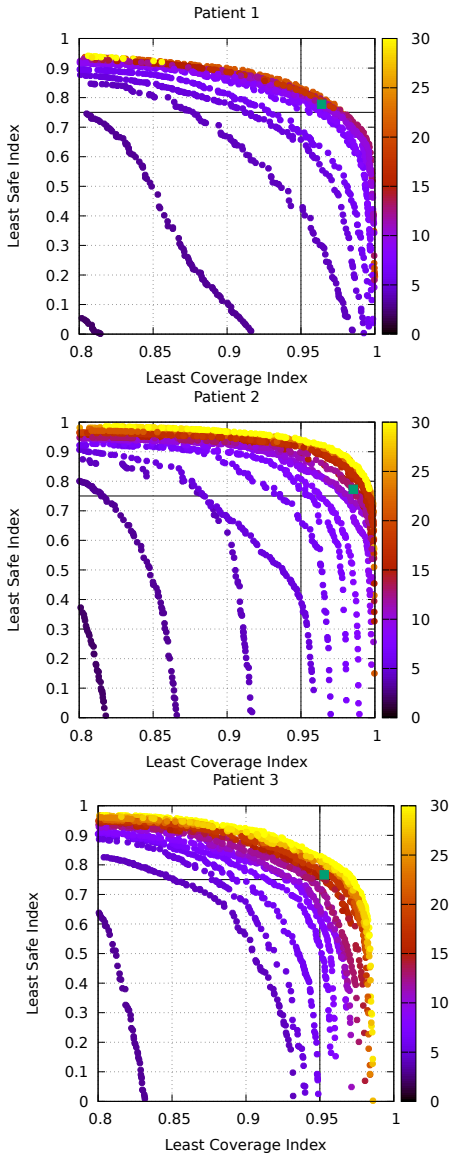


Figure 7.4: *Approximation fronts obtained with MO-GAMBIT illustrating trade-offs between three objectives for three patients. The color coding represents the number of catheters. Green points represent the high-quality solutions selected for visualization in Figure 7.5 with 10,12 and 16 catheters for patients 1,2 and 3 respectively.*

solutions lead to clearly inferior and unacceptable plans considering the other two objectives.

Outcomes still differ widely between patients. More solutions with better objective values are found for patient 2. Increasing the number of catheters for patient

2 provides bigger benefits in terms of the LCI and LSI objectives than for patient 1. This suggests that plan optimization can substantially differ for different patients, further highlighting the potential benefits of enabling insight into possible trade-offs and especially the added value of considering the number of catheters as part of optimization.

When the number of catheters is small, the addition of catheters can substantially improve the trade-offs between target coverage and organ sparing (e.g., see the gap between the 2D approximation fronts of using less vs more than 5 catheters). However, when the number of catheters is sufficient, the trade-off improvement due to adding more catheters clearly diminishes. It would be interesting for future work to investigate the marginal added value of each catheter insertion with respect to the (2D) approximation front improvement when considering a larger patient group.

The severity of trade-offs between the LCI and LSI is much higher in the more extreme regions of the approximation front. For example, at the area where LCI is larger than 0.95, trying to increase target coverage can quickly worsen the sparing of OARs. On the other hand, if a treatment plan of LCI = 0.95 is satisfiable, the healthy tissues can be considerably spared from radiation risks (e.g., compare the high value of LSI of solutions having LCI = 0.95). This is very likely part of the notorious practical hardness of deciding upon what values for a good plan to ultimately use.

Figure 7.5 illustrates selected plans obtained with MO-GAMBIT with only 10, 12 and 16 catheters for patients 1, 2 and 3 respectively. All DVH indices satisfy the clinical constraints. For these plans a doctor can decide whether it is acceptable, or whether a factor should be improved at the expense of another factor. For example, it may be desirable to use less than 12 catheters for patient 2, even with worse target coverage (but still satisfying all constraints). From the complete set of Pareto optimal solutions, a candidate plan in that trade-off direction can be taken, for which the same decision can be made, until finally the most appropriate Pareto-optimal solution is chosen without optimization needing to be re-ran.

7.5 Discussion and Conclusions

This chapter considers BT treatment pre-planning optimization from a multi-objective perspective, while combining, for the first time, a true multi-objective optimization approach with optimization directly on clinical evaluation criteria and considering trade-offs between key objectives: the covering of targets, the sparing of organs at risk, and the number of catheters. In current medical practice devising such plans depends heavily on the expertise of doctors. We wish to assist them by enabling the ability to view and consider a set of approximated Pareto optimal solutions with respect to the aforementioned objectives. With this ability, doctors may gain additional insight into possible treatment plans, potentially resulting in improved patient care. As a first proof of concept to this end, we combined real-world clinical software with a novel extension of GAMBIT to multi-objective optimization.

The generated three-objective approximation fronts for real-world patients provided a useful overview of possible treatment plans with respect to the number of catheters used, potential risk to healthy organs, and the coverage of the targets. In-

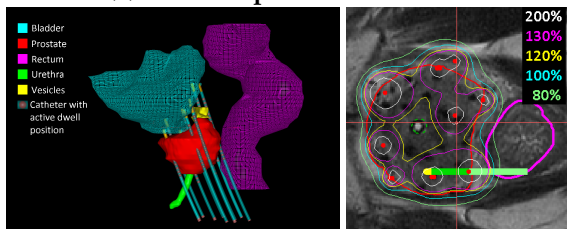
cluding, for the first time, the number and location of possible catheters placements with a multi-objective optimization approach allowed us to gain new, clinically relevant insight into BT pre-planning optimization. We were able to obtain sets of clinically acceptable non-dominated solutions, and indicated that it appears possible to obtain good solutions with smaller numbers of catheters compared to the clinically used plans for the tested patients. Using a larger number of catheters can lead to better, clinically acceptable solutions in terms of the other objectives, but the added value in key DVH indices used to evaluate plans strongly diminishes when using more than around 10 catheters. Outcomes further present insights into trade-offs between other key objectives, allowing doctors to utilize their expertise, and consider which of such trade-offs may be beneficial to the patient without re-running optimization.

Our approach based on MO-GAMBIT provides a new, promising and insightful approach for optimization of BT treatment plans, but further improvements to this approach are needed and possible. Optimization mechanisms which consider problem structure specifics such as evaluation decomposability could potentially improve optimization speed substantially. Combined with other problem-specific mechanisms, such as a multi-resolution approach to increasing the accuracy of dose calculations over time, a many-fold speedup may be obtained, reducing required run times to minutes rather than hours (even though in the pre-planning phase this is far less of an issue than when making treatment plans after physical catheter placement has taken place) and allowing for more exhaustive exploration of the search space, potentially leading to discovery of even more high-quality non-dominated solutions.

Acknowledgements

This work is part of the research programmes NWO Vrije Competitie (project number 612.001.111) and IPPSI-TA (project number 628.006.003) which are financed by the Netherlands Organisation for Scientific Research (NWO). The project in the IPPSI-TA programme is co-financed by Elekta.

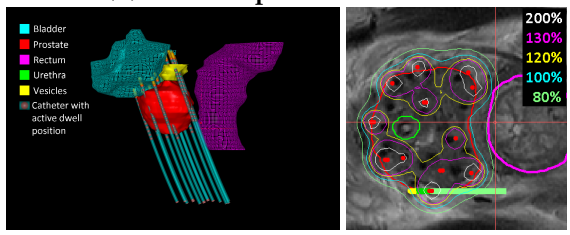
(a) Selected patient 1 solution



Prostate	Vesicles
$V_{100} = 96.39\%$	$V_{80} = 96.40\%$

Prostate	Bladder	Rectum	Urethra
$V_{150} = 27.20\%$	$V_{86} = 0.58cc$	$V_{78} = 0.78cc$	$V_{110} = 0.09cc$
$V_{200} = 11.05\%$	$V_{74} = 1.78cc$	$V_{74} = 1.09cc$	

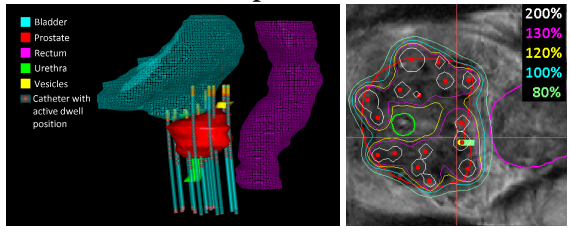
(b) Selected patient 2 solution



Prostate	Vesicles
$V_{100} = 98.53\%$	$V_{80} = 99.00\%$

Prostate	Bladder	Rectum	Urethra
$V_{150} = 19.72\%$	$V_{86} = 0.52cc$	$V_{78} = 0.81cc$	$V_{110} = 0.08cc$
$V_{200} = 6.76\%$	$V_{74} = 1.82cc$	$V_{74} = 1.21cc$	

(c) Selected patient 3 solution



Prostate	Vesicles
$V_{100} = 95.29\%$	$V_{80} = 95.50\%$

Prostate	Bladder	Rectum	Urethra
$V_{150} = 27.24\%$	$V_{86} = 0.85cc$	$V_{78} = 0.50cc$	$V_{110} = 0.09cc$
$V_{200} = 8.35\%$	$V_{74} = 1.87cc$	$V_{74} = 0.69cc$	

Figure 7.5: *Selected solutions from the approximation fronts of patients* 1, 2 *and* 3 *resulting in a high-quality plan with* 10, 12 *and* 16 *catheters, respectively. The figures on the right show the corresponding isodose lines in selected slices on the corresponding MRI scans, where red dots are active dwell positions. The tables show the corresponding DVH indices. All DVH index values satisfy the clinical constraints.*

8 Summary, Discussion and Conclusions

In this final chapter we revisit and summarize our research, results and contributions presented throughout this thesis. Further, we reflect on the work we have done and provide a discussion on potential shortcomings, possible future work as well as more general observations and lessons learned.

8.1 Summary

In this thesis we have focused on learning and exploiting problem variable dependencies when solving optimization problems with both discrete and continuous variables in a black-box setting, i.e., where these dependencies were not known a-priori. Model-Based Evolutionary Algorithms (MBEAs) are interesting in this regard as they have proven successful in overcoming non-trivial variable dependencies in case of problems with one type of variable.

We specifically looked at the Gene-pool Optimal Mixing Evolutionary Algorithms (GOMEAs) which are a relatively recent representative of MBEAs which introduce a new means of defining and exploiting important building blocks of a problem. One of our main objectives was to expand the applicability of model-based EAs to new problems and to new domains. To this end, in Chapter 3 we presented how variable dependencies can be utilized for solving maximum satisfiability (max-sat) problems better, a well-known combinatorial optimization problem that was not considered before in the literature on MBEAs. Using GOMEA-based approaches in a black-box, as well as a white-box setting, we were able to obtain good results on challenging max-sat benchmarks and showcase the importance and capabilities of learning and exploiting problem structure in terms of variable dependencies. Additionally, it is worth noting that our work on this topic led to a small but consequential change in GOMEA-based implementations going forth. Changing the new solution acceptance condition from only accepting better than parent solutions into the population to also allowing equally good solutions to replace parent solutions drastically improved the diversity of a population without loss of solution quality. For problems such as max-sat, where plateaus in the fitness landscape are commonly occurring, this allows for much better exploration of the search space, and less risk of convergence to local optima, ultimately leading to better results in the same amount of time.

The remaining core of this thesis focused on bringing the MBEA paradigm to the very challenging mixed-integer domain, where discrete and continuous variables exist simultaneously. In Chapter 4 we devised and motivated a new set of benchmark problems, which incorporate various degrees of variable dependencies and problem sizes. Our benchmark problems, derived from well-known test functions in the discrete and continuous domains, enabled us to create mixed-integer problems representing different types of problem difficulty, and different problem composition in terms of size and variable type proportions.

In Chapter 5 we introduced a novel MBEA for single-objective mixed integer optimization in the black-box setting: the Genetic Algorithm for Model-Based mixed-Integer opTimization, or GAMBIT. In GAMBIT, we introduced a balanced dependency learning and exploitation mechanism which utilized techniques already known from MBEAs in the discrete and continuous domains in an interleaved fashion for handling intra-variable dependencies. A inter-variable dependency learning and sampling mechanism was furthermore introduced to directly deal with potential dependencies between discrete and continuous problem variables. A population-level clustering mechanism allowed GAMBIT to efficiently explore different areas of the search space, while implementation of an interleaved multi-start scheme enabled out-of-the-box usage, as all algorithm specific parameters such as number of clusters or population sizes do not have to be set by a user as they are dynamic and handled internally. We compared the performance of GAMBIT against EAs that we extended from other domains into the mixed-integer domain. We also considered alternative open-source and commercial algorithms explicitly designed for mixed-integer optimization. We observed that many existing algorithms for the mixed-integer domain are very efficient for problems with many constraints. However, when faced with benchmarks where problem variables are highly dependent, GAMBIT outperforms all considered alternatives. This indicated that dependency learning and processing in the mixed-integer search space has clear added value for solving difficult problems, and GAMBIT provides some of the foundations to this end that were not utilized before for optimization of difficult, mixed-integer problems.

In Chapter 6 we expanded our dependency learning and processing approach realized with GAMBIT and considered an additional optimization challenge: multiple objectives (MO). MO-GAMBIT was proposed in which new components were introduced to address MO-specific challenges. In particular, MO-GAMBIT uses an elitist archive, a scaling mechanism to assure diversity and sampling approaches custom to the MO domain. Our results show that these dedicated mechanisms lead to superior performance compared to more standard approaches including single-objective GAMBIT-based optimization of objective scalarizations and other tested algorithms dedicated for the MO-domain.

Finally, in Chapter 7 we applied MO-GAMBIT to the challenging real-world optimization problem associated with the pre-planning stage of prostate cancer treatment with HDR brachytherapy (BT). With HDR BT, a radioactive source is passed through a series of catheters that are surgically implanted to pass through the prostate. Radiation is emitted for different time durations at different spots along these catheters. By defining the problem as a multi-objective one, we attempted to obtain new insight into tradeoffs associated with increasing coverage of the cancer-

ous areas in terms of radiation dose delivered there, while minimizing the exposure of the healthy organs. Finally, we also wished to minimize the numbers of catheters used. Starting with the MO-GAMBIT implementation, and augmenting it with some problem-specific knowledge, we obtained sets of clinically acceptable non-dominated solutions, that indicated that it is possible to obtain good solutions in terms of several commonly used clinical evaluation criteria with smaller numbers of catheters compared to the clinically used plans for the tested patients.

8.2 Discussion

While we have shown promising results of our MBEA known as GAMBIT for solving mixed-integer optimization problems, some limitations have become apparent. We briefly reflect on our observations regarding such potential issues as well as potential improvements in the following sections.

8.2.1 Learning Overhead

Working under a strict black-box assumption necessitates building algorithms that are well extendable and adaptable to various new problems and domains because little assumptions besides expecting some form of exploitable structure are made about the optimization problem. This also promotes the design of algorithms to be available 'out-of-the-box', without the need to specify any parameters or have in-depth knowledge of the problem at hand. There is however a cost to pay for such flexibility: efficiency. The overhead needed for self-adaptation of parameter settings and dependency learning to be able to tackle problems with a wide range of problem difficulty, negatively impacts the performance of the algorithm when facing problems which are relatively simpler. However, given that it is likely that many problems in the real-world, especially in the MI domain, contain some degree of inter- and intra-variable dependencies, the price of this overhead is often well worth it. It should however be considered that when facing relatively simple problems and when efficiency in terms of runtime or number of evaluations is needed, alternative approaches may be better suitable. Designing novel mechanisms to explicitly detect classes of difficulty and adapting the level of dependency exploitation accordingly could lower the necessary overhead.

8.2.2 Black-box Setting

Another point worth discussing is the real-world usability of a strict black-box approach. The ability to perform out-of-the-box optimization can be very appealing, as it allows for flexible applicability to real-world problems. It is worth considering however, that many real-world problems are not completely a black box. While a fully black-box approach may be a great starting point for optimizing such problems, utilizing some known problem properties or structure may be very beneficial. This is for instance apparent in the BT pre-planning optimization. With GAMBIT, some

problem-specific information can be used to augment the strict black-box approach, yielding greatly improved results.

Introducing such augmentations can be considered to result in a gray-box setting. One can then include, among others, reasonable initialization ranges for different variables, infusing the model with known dependencies (i.e., by including them as pre-existing FOS sets) or focusing most optimization efforts on the domain which is known to be more difficult.

Real-world mixed-integer problems can be heavily constrained. In Chapter 4 we augmented the implementation of GAMBIT with some known constraint handling mechanisms, such as the penalty method. This is a reasonable approach under the black-box assumption. However, it is also reasonable to assume that in many real-world cases the existing constraints are well established and known. Many open-source and commercial algorithms utilize the knowledge and understanding of constraints which allows them to achieve more efficient optimization. Incorporating similar considerations with GAMBIT, for instance through constraint-specific repair operators, in case problem constraints are assumed to be explicitly known, could greatly improve performance on heavily constrained problems.

8.2.3 Fitness Function

For difficult problems with expensive evaluation functions, obtaining good results comes with a very large overhead in terms of computational time due to the number of evaluations used, even though it is often less than competing approaches. The reason for this is the fact that the EAs are designed (and tested) to overcome many types of problem difficulty and to find near-optimal solutions rather than obtaining good solutions as quickly as possible, even if this comes at the cost of substantial overhead, i.e., as in expensive optimization approaches like Bayesian Optimization. Of course multi-threading and distribution of the processing across many cores can alleviate some of these issues, as is possible for many EAs due to the use of a population [19] [85] [3] [65]. Other approaches may also be possible depending on how much is known about the problem. In the current implementation of GAMBIT the evaluation function is treated as given, and the evaluation time for each such function is a necessity for any type of change to a solution, even if only few variables are concerned. In many cases, this does not have to be the case, however. The ability to leverage partial evaluations when some of the evaluation function properties are known could be of paramount importance in improving the results in terms of execution time. The impact thereof was recently shown in the Real-valued GOMEA (RV-GOMEA), which achieves great improvements in terms of scalability for problems which allow partial evaluations [20]. Bringing similar mechanisms into the mixed-integer domain could positively impact scalability for many applications.

8.3 Conclusion

Optimization is a vast field where extensive research has been actively performed for many decades in many areas. In this thesis we looked for inspiration in areas of

discrete and continuous optimization with EAs, in order to build novel competent EAs for the mixed-integer domain. The flexibility of model-based evolutionary algorithms (MBEAs) allowed us to create new MBEAs in this challenging domain, while grounding our approach to the black-box setting helped us create a general rather than a problem-specific approach.

Through our work in this thesis we have gained important new insights and understanding of how model-based EAs can be designed for and be applied in the mixed-integer domain. While mixed-integer optimization remains challenging in general, we have identified and developed critical mechanisms which allow GAMBIT, and by extension potentially other MBEAs, to be scalable and powerful alternatives in the MI optimization domain. Integrated models mechanisms allowing for obtaining a good balance between sampling discrete and continuous domains across different problem compositions, clustering mechanisms allowing for better exploration and exploitation of the problem landscape, mixed-dependency learning and model building mechanisms allowing for capturing and exploiting inter-variable dependencies, interleaved multi-restart schemes removing the need to specify any external parameters and customized MO optimization mechanisms, are some of these critical mechanisms. The particular mechanisms we proposed, and GAMBIT in general, lay down a new foundation for real-world MI optimization with MBEAs, that improves upon previous approaches from the EA field.

Based on the results obtained in this thesis, we believe that (MO-)GAMBIT, through its new and state-of-the-art approach to learning and exploiting various dependency types, provides a new competent evolutionary approach to solving black-box optimization problems with both discrete and continuous variables, thereby significantly expanding the types of problems that can be effectively handled by model-based EAs. Experimental validation of this was obtained across a range of problems, including a difficult real-world application. We believe this line of research offers a lot of new possibilities to move forward, but it is by no means complete. We particularly believe that the foundations of MI optimization with dependency-exploiting model-based EAs, provided in this thesis through GAMBIT, can serve as a motivation and inspiration for further work on model-based EAs for problems with both discrete and continuous variables.

Bibliography

[1] T. Achterberg. SCIP: Solving Constraint Integer Programs. *Mathematical Programming Computation*, 1(1):1–41, 2009.

[2] C. Adjiman, I. Androulakis, and C. Floudas. Global optimization of mixed-integer nonlinear problems. *AIChE Journal*, 46(9):1769–1797, 2000.

[3] E. Alba. Parallel evolutionary algorithms can achieve super-linear performance. *Evolutionary Computation*, 82(1):7–13, 2002.

[4] C. Audet and J. E. Dennis, Jr. Mesh Adaptive Direct Search Algorithms for Constrained Optimization. *SIAM Journal on Optimization*, 17(1):188–217, 2006.

[5] M. S. Bazaraa. *Nonlinear Programming: Theory and Algorithms*. Wiley Publishing, 3rd edition, 2013.

[6] P. Belotti. Couenne: a User's Manual. *Technical Report, Lehigh University*, 2009.

[7] M. Benichou, J. M. Gauthier, P. Girodet, G. Hentges, G. Ribiere, and O. Vincent. Experiments in mixed-integer linear programming. *Mathematical Programming*, 1(1):76–94, 1971.

[8] T. Benoist, B. Estellon, F. Gardi, R. Megel, and K. Nouioua. Localsolver 1. x: a Black-Box Local-Search Solver for 0-1 Programming. *4OR*, 9(3):299–316, 2011.

[9] P. Bonami and J. Lee. BONMIN User Manual. 2007.

[10] P. Bosman and T. Alderliesten. Incremental gaussian model-building in multi-objective edas with an application to deformable image registration. In *Proceedings of the 14th Annual Conference on Genetic and Evolutionary Computation*, GECCO '12, pages 241–248, New York, NY, USA, 2012. ACM.

[11] P. Bosman, J. Grahl, and D. Thierens. Benchmarking Parameter-Free AMaLGaM on Functions With and Without Noise. *Evolutionary Computation*, 21(3):445–469, 2013.

[12] P. Bosman and D. Thierens. The balance between proximity and diversity in multiobjective evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 7(2):174–188, 2003.

[13] P. A. Bosman. The Anticipated Mean Shift and Cluster Registration in Mixture-based EDAs for Multi-Objective Optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO'10, pages 351–358, New York, NY, USA, 2010. ACM.

[14] P. A. Bosman. The Anticipated Mean Shift and Cluster Registration in Mixture-based EDAs for Multi-objective Optimization. In *Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation*, GECCO '10, pages 351–358, New York, NY, USA, 2010. ACM.

[15] P. A. N. Bosman, J. Grahl, and D. Thierens. Enhancing the Performance of Maximum-Likelihood Gaussian EDAs Using Anticipated Mean Shift. In *Parallel Problem Solving from Nature - PPSN X*, LNCS, pages 133–143, 2008.

[16] P. A. N. Bosman, J. Grahl, and D. Thierens. AMaLGaM IDEAs in Noiseless Black-Box Optimization Benchmarking. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '09, pages 2247–2254, New York, NY, USA, 2009. ACM.

[17] P. A. N. Bosman, J. Grahl, and D. Thierens. Benchmarking parameter-free amalgam on functions with and without noise. *Evolutionary Computation*, 21(3):445–469, 2013.

[18] P. A. N. Bosman and D. Thierens. Linkage Neighbors, Optimal Mixing and Forced Improvements in Genetic Algorithms. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '12, pages 585–592, New York, NY, USA, 2012. ACM.

[19] A. Bouter, T. Alderliesten, A. Bel, C. Witteveen, and P. A. N. Bosman. Large-scale Parallelization of Partial Evaluations in Evolutionary Algorithms for Real-world Problems. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '18, pages 1199–1206, New York, NY, USA, 2018. ACM.

[20] A. Bouter, T. Alderliesten, C. Witteveen, and P. A. N. Bosman. Exploiting Linkage Information in Real-valued Optimization with the Real-valued Gene-pool Optimal Mixing Evolutionary Algorithm. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '17, pages 705–712, New York, NY, USA, 2017. ACM.

[21] S. Burer and A. N. Letchford. Non-convex mixed-integer nonlinear programming: A survey. *Surveys in Operations Research and Management Science*, 17(2):97–106, 2012.

[22] M. Bussieck and A. Meeraus. General Algebraic Modeling System (GAMS). In *Modeling Languages in Mathematical Optimization*, volume 88 of *Applied Optimization*, pages 137–157. Springer US, 2004.

[23] M. Bussieck and A. Pruessner. Mixed-integer Nonlinear Programming. *SIAG/OPT Viewsand-News*, 14(1):19–22, 2003.

[24] M. R. Bussieck and A. Drud. SBB: A New Solver for Mixed Integer Nonlinear Programming. *Talk, OR*, 2001.

[25] R. H. Byrd, J. Nocedal, and R. A. Waltz. KNITRO: An integrated package for nonlinear optimization. In *Large-scale nonlinear optimization*, pages 35–59. Springer, 2006.

[26] C. A. Coello Coello. An Introduction to Evolutionary Algorithms and Their Applications. *Advanced Distributed Systems*, pages 425–442, 2005.

[27] T. M. Cover and J. A. Thomas. *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, New York, NY, USA, 2006.

[28] K. Deb. *Multi-Objective Optimization Using Evolutionary Algorithms*. John Wiley & Sons, Inc., NY, USA, 2001.

[29] K. Deb and D. E. Goldberg. Analyzing Deception in Trap Functions. *Foundations of Genetic Algorithms*, 2:93 – 108, 1993.

[30] T. Deist and B. Gorissen. High-dose-rate prostate brachytherapy inverse planning on dose-volume criteria by simulated annealing. *Physics in Medicine and Biology*, 61(3):1155, 2016.

[31] L. Eapen, C. Kayser, Y. Deshaies, G. Perry, E. Choan, C. Morash, J. Cygler, D. Wilkins, and S. Dahrouge. Correlating the degree of needle trauma during prostate brachytherapy and the development of acute urinary toxicity. *International Journal of Radiation Oncology*, 59(5):1392–1394, 2004.

[32] A. E. Eiben and C. A. Schippers. On evolutionary exploration and exploitation. *Fundam. Inf.*, 35(1-4):35–50, 1998.

[33] G. F. Pulse Code Communication. In *United States Patent Number 2632058*, 1953.

[34] D. B. Fogel. Evolutionary Computation: Toward a New Philosophy of Machine Intelligence. *IEEE Press Series on Computational Intelligence*, 2006.

[35] G. Gamrath, T. Fischer, T. Gally, A. M. Gleixner, G. Hendel, T. Koch, S. J. Maher, M. Miltenberger, B. Müller, M. E. Pfetsch, C. Puchert, D. Rehfeldt, S. Schenker, R. Schwarz, F. Serrano, Y. Shinano, S. Vigerske, D. Weninger, M. Winkler, J. T. Witt, and J. Witzig. The SCIP Optimization Suite 3.2. Technical Report 15-60, ZIB, Takustr.7, 14195 Berlin, 2016.

[36] D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1st edition, 1989.

[37] I. Gronau and S. Moran. Optimal Implementations of UPGMA and Other Common Clustering Algorithms. *Information Processing Letters*, 104(9):205–210, 2007.

[38] I. Grosse, P. Bernaola-Galvan, P. Carpena, R. Roman Roldan, J. Oliver, and H. Stanley. Analysis of symbolic sequences using the jensen-shannon divergence. *Physical review. E, Statistical, nonlinear, and soft matter physics*, 65:041905, 2002.

[39] I. E. Grossmann. Review of nonlinear mixed-integer and disjunctive programming techniques. *Optimization and Engineering*, 3(3):227–252, 2002.

[40] I. E. Grossmann and Z. Kravanja. Mixed-integer nonlinear programming: A survey of algorithms and applications. *Large-Scale Optimization with Applications: Part II: Optimal Design and Control*, pages 73–100, 1997.

[41] I. E. Grossmann, J. Viswanathan, A. Vecchietti, R. Raman, E. Kalvelagen, et al. Gams/DICOPT: A discrete continuous optimization package. *GAMS Corporation Inc*, 2002.

[42] N. Hansen, S. Finck, R. Ros, and A. Auger. Benchmarking 2009: Noiseless Functions Definitions. RR-6829. *INRIA*, 2009.

[43] G. Harik, E. Cantu-Paz, D. E. Goldberg, and B. L. Miller. The gambler's ruin problem, genetic algorithms, and the sizing of populations. *Evolutionary computation*, 7:231–53, 1999.

[44] G. R. Harik and F. G. Lobo. A Parameter-Less Genetic Algorithm. *IEEE transactions on evolutionary computation*, pages 523–528, 1999.

[45] J. H. Holland. Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence. MIT Press, 1975.

[46] J. H. Holland. Genetic Algorithms. *Scientific American*, 267(1):66–73, 1992.

[47] H. H. Hoos and T. Stützle. SATLIB: An Online Resource for Research on SAT. pages 283–292. IOS Press, 2000.

[48] P. J. Hoskin, A. Colombo, A. Henry, P. Niehoff, T. P. Hellebust, F.-A. Siebert, and G. Kovacs. Gec/estro recommendations on high dose rate afterloading brachytherapy for localised prostate cancer: An update. *Radiotherapy and Oncology*, 107(3):325 – 332, 2013.

[49] S. Kang, R. Chou, R. Dodge, R. Clough, H. Kang, M. Bowen, B. Steffey, S. Das, S. Zhou, A. Whitehurst, et al. Acute urinary toxicity following transperineal prostate brachytherapy using a modified quimby loading method. *International Journal of Radiation Oncology*, 50(4):937–945, 2001.

[50] A. Karabis, P. Belotti, and D. Baltas. *Optimization of Catheter Position and Dwell Time in Prostate HDR Brachytherapy using HIPO and Linear Programming*, pages 612–615. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[51] J. Knowles and D. Corne. On metrics for comparing nondominated sets. In *Proceedings of the 2002 Congress on Evolutionary Computation*, volume 1, pages 711–716, May 2002.

[52] A. Kraskov, H. Stögbauer, and P. Grassberger. Estimating mutual information. *Physical review. E, Statistical, nonlinear, and soft matter physics*, 69:066138, 2004.

[53] A. Kraskov, H. Stögbauer, and P. Grassberger. Estimating Mutual Information. *Phys. Rev. E*, 69, 2004.

[54] H. W. Kuhn and B. Yaw. The Hungarian method for the assignment problem. *Naval Res. Logist. Quart*, pages 83–97, 1955.

[55] P. Larranaga and J. A. Lozano. *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*. 2002.

[56] S. Le Digabel. Algorithm 909: NOMAD: Nonlinear Optimization with the MADS Algorithm. *ACM Trans. Math. Softw.*, 37(4):44:1–44:15, 2011.

[57] R. Li, M. T. M. Emmerich, J. Eggermont, T. Bäck, M. Schütz, J. Dijkstra, and J. H. C. Reiber. Mixed Integer Evolution Strategies for Parameter Optimization. *Evolutionary Computation*, 21(1):29–64, 2013.

[58] J. Lin. Divergence measures based on the shannon entropy. *IEEE Transactions on Information Theory*, 37(1):145–151, Jan 1991.

[59] Y. Lin and L. Schrage. The global solver in the LINDO API. *Optimization Methods & Software*, 24(4-5):657–668, 2009.

[60] J. Lozano, P. Larranaga, I. Inza, and E. Bengoetxea. Towards a new evolutionary computation. Advances on estimation of distribution algorithms. volume 192, 2006.

[61] H. Luong and P. Bosman. Elitist archiving for multi-objective evolutionary algorithms: To adapt or not to adapt. In *International Conference on Parallel Problem Solving from Nature*, pages 72–81. Springer, 2012.

[62] N. H. Luong, H. La Poutré, and P. A. Bosman. Multi-objective Gene-pool Optimal Mixing Evolutionary Algorithms. In *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation*, GECCO '14, pages 357–364, New York, NY, USA, 2014. ACM.

[63] N. H. Luong, H. La Poutré, and P. A. Bosman. Exploiting Linkage Information and Problem-Specific Knowledge in Evolutionary Distribution Network Expansion Planning. In *Proceedings of the 2015 on Genetic and Evolutionary Computation Conference*, GECCO '15, pages 1231–1238, New York, NY, USA, 2015. ACM.

[64] N. H. Luong, H. L. Poutré, and P. A. Bosman. Multi-objective Gene-pool Optimal Mixing Evolutionary Algorithm with the Interleaved Multi-start scheme. volume 40, pages 238 – 254, 2018.

[65] G. Luque and E. Alba. Parallel genetic algorithms: Theory and realworld applications. volume 367, pages 1–183, 2011.

[66] G. M. The Binary Gray Code. In *Knotted Doughnuts and Other Mathematical Entertainments*, New York, NY, USA, 1986. Freeman.

[67] D. C. R. managed by IKNL © January 2017. *Prostaatkanker (Prostaatcarcinoom)*. http://www.prostaat.nl/aandoeningen/prostaatkanker-prostaatcarcinoom/.

[68] H. Mühlenbein and G. Paaß. *From recombination of genes to the estimation of distributions I. Binary parameters*, pages 178–187. Springer Berlin Heidelberg, Berlin, Heidelberg, 1996.

[69] M. Pelikan and D. E. Goldberg. Hierarchical BOA Solves Ising Spin Glasses and MAXSAT. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '03, pages 1271–1282. Springer, 2003.

[70] M. Pelikan, D. E. Goldberg, and E. Cantu-Paz. BOA: The Bayesian Optimization Algorithm. pages 525–532. Morgan Kaufmann, 1999.

[71] A. Prügel-Bennett and M. Tayarani-Najaran. Maximum satisfiability: Anatomy of the fitness landscape for a hard combinatorial optimization problem. *IEEE transactions on evolutionary computation*, 16(3):319–338, 2012.

[72] S. B. Rana and L. D. Whitley. Genetic Algorithm Behavior in the MAXSAT Domain. In *Proceedings of the 5th International Conference on Parallel Problem Solving from Nature*, Parallel Problem Solving from Nature - PPSN V, pages 785–794, London, UK, UK, 1998. Springer-Verlag.

[73] B. C. Ross. Mutual Information between Discrete and Continuous Data Sets. *PLoS ONE*, 9(2), 2014.

[74] T. Runarsson and X. Yao. Constrained Evolutionary Optimization. In *Evolutionary Optimization*, volume 48 of *International Series in Operations Research and Management Science*, pages 87–113. Springer US, 2002.

[75] K. L. Sadowski, P. Bosman, and D. Thierens. Learning and Exploiting Mixed Variable Dependencies with a Model-Based EA. In *IEEE Congress on Evolutionary Computation (CEC'16)*, pages 4382–4389, 2016.

[76] K. L. Sadowski, P. A. Bosman, and D. Thierens. A Clustering-Based Model-Building EA for Optimization Problems with Binary and Real-Valued Variables. In *Proceedings of the 2015 on Genetic and Evolutionary Computation Conference*, GECCO '15, pages 911–918, New York, NY, USA, 2015. ACM.

[77] K. L. Sadowski, P. A. N. Bosman, and D. Thierens. On the Usefulness of Linkage Processing for Solving MAX-SAT. In *Genetic and Evolutionary Computation Conference, GECCO '13, Amsterdam, The Netherlands, July 6-10, 2013*, pages 853–860, 2013.

[78] K. L. Sadowski, D. Thierens, and P. A. N. Bosman. Combining Model-based EAs for Mixed-Integer Problems. In *Parallel Problem Solving from Nature - PPSN XIII*, pages 342–351, 2014.

[79] K. L. Sadowski, D. Thierens, and P. A. N. Bosman. GAMBIT: A Parameterless Model-Based Evolutionary Algorithm for Mixed-Integer Problems. *Evolutionary Computation*, 26(1):117–143, 2018.

[80] K. L. Sadowski, M. C. van der Meer, N. H. Luong, T. Alderliesten, D. Thierens, R. van der Laarse, Y. Niatsetski, A. Bel, and P. A. N. Bosman. Exploring Trade-offs Between Target Coverage, Healthy Tissue Sparing, and the Placement of Catheters in HDR Brachytherapy for Prostate Cancer Using a Novel Multi-objective Model-based Mixed-integer Evolutionary Algorithm. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '17, pages 1224–1231, New York, NY, USA, 2017. ACM.

[81] N. V. Sahinidis. Mixed-integer nonlinear programming 2018. *Optimization and Engineering*, 20(2):301–306, 2019.

[82] B. Selman, H. Kautz, and B. Cohen. Local search strategies for satisfiability testing. In *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 521–532, 1995.

[83] B. Selman, H. Levesque, and D. Mitchell. A new method for solving hard satisfiability problems. In *Proceedings of AAAI-92*, pages 440–446, 1992.

[84] R. R. Sokal and C. D. Michener. A statistical method for evaluating systematic relationships. *University of Kansas Science Bulletin*, 38:1409–1438, 1958.

[85] D. Sudholt. Parallel evolutionary algorithms. In *Springer Handbook of Computational Intelligence*, pages 929–959, Berlin, Heidelberg, 2015. Springer Berlin Heidelberg.

[86] D. Thierens. Scalability Problems of Simple Genetic Algorithms. *Evolutionary Computation*, 7:331–52, 1999.

[87] D. Thierens. The Linkage Tree Genetic Algorithm. In *Parallel Problem Solving from Nature - PPSN XI*, LNCS, pages 264–273, Berlin, 2010. Springer–Verlag.

[88] D. Thierens and P. A. N. Bosman. Optimal Mixing Evolutionary Algorithms. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '11, pages 617–624, New York, NY, USA, 2011. ACM.

[89] D. Thierens and P. A. N. Bosman. Predetermined versus Learned Linkage Models. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '12, pages 289–296, New York, NY, USA, 2012. ACM.

[90] C. Vargas, M. Ghilezan, M. Hollander, G. Gustafson, H. Korman, J. Gonzalez, and A. Martinez. A new model using number of needles and androgen deprivation to predict chronic urinary toxicity for high or low dose rate prostate brachytherapy. *The Journal of urology*, 174(3):882–887, 2005.

[91] M. Črepinšek, S.-H. Liu, and M. Mernik. Exploration and exploitation in evolutionary algorithms: A survey. volume 45, pages 35:1–35:33, New York, NY, USA, 2013. ACM.

[92] T. Westerlund and K. Lundqvist. *Alpha-ECP, Version 5.01: An interactive MINLP-Solver Based on the Extended Cutting Plane Method*. Åbo Akademi, 2001.

Samenvatting

In dit proefschrift hebben we ons gericht op het leren en benutten van afhankelijkheden tussen probleemvariabelen bij het oplossen van optimalisatieproblemen in een black-box setting, d.w.z., in een situatie waarin deze afhankelijkheden niet a priori bekend waren. De belangrijkste bron van moeilijkheden voor dergelijke problemen houdt nauw verband met schaalbaarheid en doet zich vaak voor in alledaagse situaties. Model-Building Evolutionary Algorithms (MBEA's, modelgebaseerde evolutionaire algoritmen) zijn in dit geval interessant omdat ze succesvol zijn gebleken bij het omgaan met niet-triviale variabele-afhankelijkheden. In dit proefschrift kijken we specifiek naar de Gene-pool Optimal Mixing Evolutionary Algorithms (GOMEA's, genenpoel optimaal mengende evolutionaire algoritmen), die een relatief recente soort MBEA's zijn. Deze introduceren een nieuwe manier voor het definiëren en benutten van belangrijke bouwstenen van een probleem. Een van onze hoofddoelen was het uitbreiden van de toepasbaarheid van MBEA's op nieuwe problemen en nieuwe domeinen. Daartoe presenteren we in hoofdstuk 3 hoe afhankelijkheden tussen variabelen max-sat, een bekend combinatorisch optimalisatieprobleem dat nog niet eerder in de literatuur over MBEA's werd overwogen. Met behulp van GOMEA-gebaseerde algoritmen in een black-box- en in een white-box-setting, behaalden we goede resultaten op uitdagende max-sat-benchmarks. Hiermee laten we de mogelijkheden van het leren en benutten van probleemstructuren in termen van afhankelijkheden tussen variabelen zien.

In het vervolg van het proefschrift richten we ons op het toepassen van het MBEA-paradigma op het uitdagende gemengd geheeltallige domein, waarin zowel discrete als continue variabelen voorkomen in een probleem. In hoofdstuk 4 presenteren en onderbouwen we een nieuwe verzameling benchmarks, die verschillende maten van afhankelijkheden tussen variabelen en probleemcomplexiteit met zich meebrengen. Onze benchmarkproblemen, die zijn afgeleid van bekende functies in de discrete en continue domeinen, stellen ons in staat om gemengd geheeltallige problemen zo aan te passen dat ze verschillende soorten probleemcomplexiteiten vertegenwoordigen. Met deze verzameling flexibele benchmarks, waarmee we verschillende afhankelijkheidstypen en probleemsamenstellingen kunnen uitdrukken in termen van grootte en verhoudingen tussen soorten variabelen, waren we in staat om nieuwe manieren te introduceren om optimalisatie te bewerkstelligen in het geval van problemen met

een enkele doelfunctie alsook problemen met meerdere tegenstrijdige doelfuncties in een black-box-setting en om de prestaties ervan te vergelijken met verscheidene bestaande algoritmen.

In hoofdstuk 5 hebben we een nieuw evolutionair algoritme (EA) geïntroduceerd voor gemengd geheeltallige optimalisatie in het geval van een enkele doelfunctie in de black-box-setting: het Genetic Algorithm for Model-Based mixed-Integer opTimisation (GAMBIT, genetische algoritme voor modelgebaseerde gemengd geheeltallige optimalisatie). In GAMBIT hebben we leer- en benuttingsmechanismen geïntroduceerd die technieken uit de discrete en continue domeinen op een evenwichtige manier met elkaar verweven om goed om te kunnen gaan met afhankelijkheden tussen variabelen van hetzelfde type. Een apart leer- en samplingmechanisme voor afhankelijkheden tussen variabelen van verschillende typen werd geïntroduceerd om direct om te gaan met mogelijke afhankelijkheden tussen discrete en continue probleemvariabelen. Dankzij een clusteringmechanisme op populatieniveau kon GAMBIT verschillende gebieden van de zoekruimte efficiënt verkennen, terwijl de implementatie van een zogenoemd interleaved multi-startschema, waarbij meerdere runs van een EA met een steeds grotere populatie op een verweven manier tegelijkertijd uitgevoerd worden, het geheel direct gebruiksklaar maakte. Algoritme-specifieke parameters (zoals het aantal clusters of populatiegroottes) hoeven hierdoor niet door een gebruiker te worden ingesteld, omdat deze dynamisch zijn en intern worden verwerkt. We hebben de prestaties van GAMBIT vergeleken met EA's die we uit andere domeinen hebben uitgebreid naar het domein van gemengd geheeltallige problemen. Daarnaast, hebben we ook vergelijkingen gedaan met alternatieven: open-source en commerciële algoritmen die expliciet zijn ontworpen voor optimalisatie van gemengd geheeltallige problemen. We constateren dat veel bestaande algoritmen voor het gemengd geheeltallige domein zeer efficiënt zijn voor problemen met meerdere randvoorwaarden. Op benchmarks waar probleemvariabelen sterk afhankelijk zijn echter overtreft GAMBIT alle bekeken alternatieven. Hieruit bleek dat het leren en benutten van afhankelijkheid in het gemengd geheeltallige domein een duidelijke toegevoegde waarde heeft voor het oplossen van moeilijke gemengd geheeltallige problemen; GAMBIT biedt hiervoor fundamenten die niet eerder werden gebruikt voor het optimaliseren van complexe gemengd geheeltallige problemen (met EA's).

In hoofdstuk 6 hebben we onze manier van leren van benutten van afhankelijkheden (gerealiseerd met GAMBIT) uitgebreid en een extra optimalisatie-uitdaging onderzocht: meerdere doelfuncties (multiple objectives, MO). Het hoofdstuk introduceerde MO-GAMBIT, waarin nieuwe componenten werden aangevoerd om MO-specifieke uitdagingen aan te gaan. In het bijzonder gebruikt MO-GAMBIT een archief waarin de beste tot nog toe gevonden oplossingen opgeslagen worden. Dit is een mechanisme om diversiteit te behouden en nieuwe oplossingen te maken die specifiek geschikt zijn voor het MO-domein. Onze resultaten laten zien dat, in vergelijking met meer gangbare benaderingen, deze toegewijde mechanismen leiden tot superieure prestaties van MO-GAMBIT, waaronder het herhaaldelijk oplossen van gescalariseerde enkelvoudige doelfuncties met GAMBIT.

Tot slot hebben we in hoofdstuk 7 MO-GAMBIT toegepast op het complexe optimalisatie probleem dat komt kijken bij de pre-planningsfase van een prostaatkan-

kerbehandeling met HDR-brachytherapie (BT). Bij HDR BT wordt een radioactieve bron door een reeks chirurgisch ingebrachte katheters, die door de prostaat lopen, geleid. Straling wordt uitgezonden voor verschillende tijdsduren en op verschillende plekken in deze katheters. Door dit probleem als een probleem met meerdere doelfuncties te definiëren, hebben we getracht de afgegeven stralingsdosis te vergroten in de gebieden aangetast door kanker, terwijl de blootstelling van gezonde organen werd geminimaliseerd. Ten slotte wilden we ook het aantal gebruikte katheters minimaliseren. Beginnend met de MO-GAMBIT-implementatie, en aangevuld met probleem-specifieke kennis, verkregen we verzamelingen van klinisch aanvaardbare niet-gedomineerde oplossingen. Deze geven aan dat het mogelijk is om goede oplossingen te verkrijgen in termen van verschillende veelgebruikte klinische evaluatiecriteria met een kleiner aantal katheters dan waarmee de door ons geteste patiënten behandeld werden in de huidige klinische setting.

Op basis van de resultaten in dit proefschrift zijn we van mening dat (MO-)GAMBIT door de nieuwe aanpak van leren, alsmede het benutten van verschillende afhankelijkheidstypen, een nieuwe competente evolutionair algoritmische methode biedt voor het oplossen van black-box-optimalisatieproblemen met zowel discrete als continue variabelen. Hiermee worden de soorten problemen die effectief kunnen worden opgelost door modelgebaseerde EA's aanzienlijk uitgebreid. Experimentele validatie hiervan werd verkregen middels een breed spectrum van problemen, waaronder complexe toepassingen uit de dagelijkse praktijk. Wij zijn van mening dat we door ons onderzoek een geavanceerde nieuwe aanpak van modelgebaseerde optimalisatie met GAMBIT hebben kunnen creëren. Wij geloven dat, hoewel deze onderzoekslijn zeker niet voltooid is, ze veel nieuwe mogelijkheden voor de toekomst biedt. Daarnaast denken we dat de basis van MI-optimalisatie met afhankelijkheid-exploiterende modelgebaseerde EA's in dit proefschrift via GAMBIT kan dienen als zowel motivatie en inspiratie voor verder onderzoek naar modelgebaseerde EA's voor problemen met discrete en continue variabelen.

Publication List

K. L. Sadowski, P. A. N. Bosman, and D. Thierens. On the Usefulness of Linkage Processing for Solving MAX-SAT. In Genetic and Evolutionary Computation Conference, GECCO '13, Amsterdam, The Netherlands, July 6–10, 2013, pages 853—860, 2013.

K. L. Sadowski, D. Thierens, and P. A. N. Bosman. Combining Model-based EAs for Mixed-Integer Problems. In Parallel Problem Solving from Nature – PPSN XIII, pages 342—351, 2014.

K. L. Sadowski, D. Thierens, and P. A. N. Bosman. GAMBIT: A Parameterless Model-Based Evolutionary Algorithm for Mixed-Integer Problems. Evolutionary Computation, 26(1):117—143, 2018.

K. L. Sadowski, P. Bosman, and D. Thierens. Learning and Exploiting Mixed Variable Dependencies with a Model-Based EA. In IEEE Congress on Evolutionary Computation (CEC'16), pages 4382—4389, 2016.

K. L. Sadowski, P. A. Bosman, and D. Thierens. A Clustering-Based Model-Building EA for Optimization Problems with Binary and Real-Valued Variables. In Proceedings of the 2015 on Genetic and Evolutionary Computation Conference, GECCO '15, pages 911—918, New York, NY, USA, 2015. ACM.

K. L. Sadowski, D. Thierens, and P.A.N. Bosman. Optimization of Multi-Objective Mixed-Integer Problems with a Model-Based Evolutionary Algorithm in a Black-Box Setting. *[Submitted for review (journal article)]*

K. L. Sadowski, M. C. van der Meer, N. H. Luong, T. Alderliesten, D. Thierens, R. van der Laarse, Y. Niatsetski, A. Bel, and P. A. N. Bosman. Exploring Tradeoffs Between Target Coverage, Healthy Tissue Sparing, and the Placement of Catheters in HDR Brachytherapy for Prostate Cancer Using a Novel Multiobjective Model-based Mixed-integer Evolutionary Algorithm. In Proceedings of the Genetic and Evolutionary Computation Conference, GECCO '17, pages 1224—1231, 2017. ACM.

Acknowledgments

The publication of this thesis would not be possible without help, guidance and patience of many others. In this section I would like to mention just a few of the people that have contributed in a very major way to the completion of this work.

Firstly, I would like to extend my gratitude to my two daily supervisors: Peter Bosman and Dirk Thierens. Your continuous guidance and engagement were the cornerstone of the quality and novelty of the researched presented in this work. I truly cannot imagine a completion of this work, without your in-depth involvement. I'd further like to thank my second promotor Mehdi Dastani for his support, especially in the final stages.

I would like to thank all my co-authors, who's contributions and insights shaped the research presented in this thesis: Marjolein C van der Meer, Ngoc Hoang Luong, Tanja Alderliesten, Rob van der Laarse, Yury Niatsetski, Arjan Bel.

I would also like to acknowledge a group of colleges and friends, who's input, academic and/or motivational has been a driving force leading up to the publication of this thesis: Thomas Kraak, Bas Testerink, Marco Virgolin, Stef Maree, Anoton Bouter - my colleagues in both Utrecht University and CWI whom I engaged in may stimulating discussions on regular basis; Dave Small, Jelmer de Vries, Irfan Ilgin, Martin de Jong, Gert-Jan van der Krogt - sources of great help and inspiration. A special thank you goes out to Chiara Hübner Travezán, who designed the amazing cover of this book.

Last but not least, I could not have gotten here without the love and support of my closest family: my wife Jessica, who has been there for me every step of the way, my parents Maria and Stanislaw, and my sister Dorota, who's continuous support has been paramount not just in my academic career, but throughout my life, and my Peruvian family: Elva and Frank, Karen and John: Your continuous support has and always will be one of the major motivators in my professional and private life.

I have never been good in elaborate notes, so I shall conclude swiftly: from the bottom of my heart, thank you all!

Specjalne podziękowania skierowane są do mojej rodziny w Polsce.
W szczególności rodzicom Marii i Stanisławowi oraz siostrze Dorocie.

Bez waszej pomocy, motywacji i wsparcia,
nie tylko wydanie tej książki,
ale jakiekolwiek ważne etapy mojego życia nie byłyby możliwe.

Bardzo was kocham, i dziękuję z calego serca!

Curriculum Vitae

Education

Krzysztof has obtained a Bachelor's degree in Computer Engineering from the University of Florida in Gainesville, USA. He obtained his Master's degree in Technical Artificial Intelligence from Utrecht University in the Netherlands.

Work Experience

Between 2012 and 2017 Krzysztof conducted PhD research in a joint project between Utrecht University and CWI Amsterdam, focusing on Genetic Algorithms and Mixed-Integer optimization. Since, he worked in the private sector: first as a Senior AI Developer for Monitorilinq B.V., and later as Lead AI Engineer for Maistering B.V.

SIKS Dissertation Series

1998

1 Johan van den Akker (CWI) *DEGAS: An Active, Temporal Database of Autonomous Objects*
2 Floris Wiesman (UM) *Information Retrieval by Graphically Browsing Meta-Information*
3 Ans Steuten (TUD) *A Contribution to the Linguistic Analysis of Business Conversations*
4 Dennis Breuker (UM) *Memory versus Search in Games*
5 E. W. Oskamp (RUL) *Computerondersteuning bij Straftoemeting*

1999

1 Mark Sloof (VUA) *Physiology of Quality Change Modelling: Automated modelling of*
2 Rob Potharst (EUR) *Classification using decision trees and neural nets*
3 Don Beal (UM) *The Nature of Minimax Search*
4 Jacques Penders (UM) *The practical Art of Moving Physical Objects*
5 Aldo de Moor (KUB) *Empowering Communities: A Method for the Legitimate User-Driven*
6 Niek J. E. Wijngaards (VUA) *Re-design of compositional systems*
7 David Spelt (UT) *Verification support for object database design*
8 Jacques H. J. Lenting (UM) *Informed Gambling: Conception and Analysis of a Multi-Agent Mechanism*

2000

1 Frank Niessink (VUA) *Perspectives on Improving Software Maintenance*
2 Koen Holtman (TUe) *Prototyping of CMS Storage Management*
3 Carolien M. T. Metselaar (UvA) *Sociaal-organisatorische gevolgen van kennistechnologie*
4 Geert de Haan (VUA) *ETAG, A Formal Model of Competence Knowledge for User Interface*
5 Ruud van der Pol (UM) *Knowledge-based Query Formulation in Information Retrieval*
6 Rogier van Eijk (UU) *Programming Languages for Agent Communication*
7 Niels Peek (UU) *Decision-theoretic Planning of Clinical Patient Management*
8 Veerle Coupé (EUR) *Sensitivity Analyis of Decision-Theoretic Networks*
9 Florian Waas (CWI) *Principles of Probabilistic Query Optimization*
10 Niels Nes (CWI) *Image Database Management System Design Considerations, Algorithms and Architecture*
11 Jonas Karlsson (CWI) *Scalable Distributed Data Structures for Database Management*

2001

1 Silja Renooij (UU) *Qualitative Approaches to Quantifying Probabilistic Networks*
2 Koen Hindriks (UU) *Agent Programming Languages: Programming with Mental Models*
3 Maarten van Someren (UvA) *Learning as problem solving*
4 Evgueni Smirnov (UM) *Conjunctive and Disjunctive Version Spaces with Instance-Based Boundary Sets*
5 Jacco van Ossenbruggen (VUA) *Processing Structured Hypermedia: A Matter of Style*
6 Martijn van Welie (VUA) *Task-based User Interface Design*
7 Bastiaan Schonhage (VUA) *Diva: Architectural Perspectives on Information Visualization*
8 Pascal van Eck (VUA) *A Compositional Semantic Structure for Multi-Agent Systems Dynamics*
9 Pieter Jan 't Hoen (RUL) *Towards Distributed Development of Large Object-Oriented Models*

10 Maarten Sierhuis (UvA) *Modeling and Simulating Work Practice*
11 Tom M. van Engers (VUA) *Knowledge Management*

2002

1 Nico Lassing (VUA) *Architecture-Level Modifiability Analysis*
2 Roelof van Zwol (UT) *Modelling and searching web-based document collections*
3 Henk Ernst Blok (UT) *Database Optimization Aspects for Information Retrieval*
4 Juan Roberto Castelo Valdueza (UU) *The Discrete Acyclic Digraph Markov Model in Data Mining*
5 Radu Serban (VUA) *The Private Cyberspace Modeling Electronic*
6 Laurens Mommers (UL) *Applied legal epistemology: Building a knowledge-based ontology of*
7 Peter Boncz (CWI) *Monet: A Next-Generation DBMS Kernel For Query-Intensive*
8 Jaap Gordijn (VUA) *Value Based Requirements Engineering: Exploring Innovative*
9 Willem-Jan van den Heuvel (KUB) *Integrating Modern Business Applications with Objectified Legacy*
10 Brian Sheppard (UM) *Towards Perfect Play of Scrabble*
11 Wouter C. A. Wijngaards (VUA) *Agent Based Modelling of Dynamics: Biological and Organisational Applications*
12 Albrecht Schmidt (UvA) *Processing XML in Database Systems*
13 Hongjing Wu (TUe) *A Reference Architecture for Adaptive Hypermedia Applications*
14 Wieke de Vries (UU) *Agent Interaction: Abstract Approaches to Modelling, Programming and Verifying Multi-Agent Systems*
15 Rik Eshuis (UT) *Semantics and Verification of UML Activity Diagrams for Workflow Modelling*
16 Pieter van Langen (VUA) *The Anatomy of Design: Foundations, Models and Applications*
17 Stefan Manegold (UvA) *Understanding, Modeling, and Improving Main-Memory Database Performance*

2003

1 Heiner Stuckenschmidt (VUA) *Ontology-Based Information Sharing in Weakly Structured Environments*
2 Jan Broersen (VUA) *Modal Action Logics for Reasoning About Reactive Systems*
3 Martijn Schuemie (TUD) *Human-Computer Interaction and Presence in Virtual Reality Exposure Therapy*
4 Milan Petkovic (UT) *Content-Based Video Retrieval Supported by Database Technology*
5 Jos Lehmann (UvA) *Causation in Artificial Intelligence and Law: A modelling approach*
6 Boris van Schooten (UT) *Development and specification of virtual environments*
7 Machiel Jansen (UvA) *Formal Explorations of Knowledge Intensive Tasks*
8 Yongping Ran (UM) *Repair Based Scheduling*
9 Rens Kortmann (UM) *The resolution of visually guided behaviour*
10 Andreas Lincke (UvT) *Electronic Business Negotiation: Some experimental studies on the interaction between medium, innovation context and culture*
11 Simon Keizer (UT) *Reasoning under Uncertainty in Natural Language Dialogue using Bayesian Networks*
12 Roeland Ordelman (UT) *Dutch speech recognition in multimedia information retrieval*
13 Jeroen Donkers (UM) *Nosce Hostem: Searching with Opponent Models*
14 Stijn Hoppenbrouwers (KUN) *Freezing Language: Conceptualisation Processes across ICT-Supported Organisations*
15 Mathijs de Weerdt (TUD) *Plan Merging in Multi-Agent Systems*
16 Menzo Windhouwer (CWI) *Feature Grammar Systems: Incremental Maintenance of Indexes to Digital Media Warehouses*
17 David Jansen (UT) *Extensions of Statecharts with Probability, Time, and Stochastic Timing*
18 Levente Kocsis (UM) *Learning Search Decisions*

2004

1 Virginia Dignum (UU) *A Model for Organizational Interaction: Based on Agents, Founded in Logic*
2 Lai Xu (UvT) *Monitoring Multi-party Contracts for E-business*
3 Perry Groot (VUA) *A Theoretical and Empirical Analysis of Approximation in Symbolic Problem Solving*
4 Chris van Aart (UvA) *Organizational Principles for Multi-Agent Architectures*
5 Viara Popova (EUR) *Knowledge discovery and monotonicity*
6 Bart-Jan Hommes (TUD) *The Evaluation of Business Process Modeling Techniques*
7 Elise Boltjes (UM) *Voorbeeldig onderwijs: voorbeeldgestuurd onderwijs, een opstap naar abstract denken, vooral voor meisjes*
8 Joop Verbeek (UM) *Politie en de Nieuwe Internationale Informatiemarkt, Grensregionale politiële gegevensuitwisseling en digitale expertise*

9 Martin Caminada (VUA) *For the Sake of the Argument: explorations into argument-based reasoning*
10 Suzanne Kabel (UvA) *Knowledge-rich indexing of learning-objects*
11 Michel Klein (VUA) *Change Management for Distributed Ontologies*
12 The Duy Bui (UT) *Creating emotions and facial expressions for embodied agents*
13 Wojciech Jamroga (UT) *Using Multiple Models of Reality: On Agents who Know how to Play*
14 Paul Harrenstein (UU) *Logic in Conflict. Logical Explorations in Strategic Equilibrium*
15 Arno Knobbe (UU) *Multi-Relational Data Mining*
16 Federico Divina (VUA) *Hybrid Genetic Relational Search for Inductive Learning*
17 Mark Winands (UM) *Informed Search in Complex Games*
18 Vania Bessa Machado (UvA) *Supporting the Construction of Qualitative Knowledge Models*
19 Thijs Westerveld (UT) *Using generative probabilistic models for multimedia retrieval*
20 Madelon Evers (Nyenrode) *Learning from Design: facilitating multidisciplinary design teams*

2005

1 Floor Verdenius (UvA) *Methodological Aspects of Designing Induction-Based Applications*
2 Erik van der Werf (UM) *AI techniques for the game of Go*
3 Franc Grootjen (RUN) *A Pragmatic Approach to the Conceptualisation of Language*
4 Nirvana Meratnia (UT) *Towards Database Support for Moving Object data*
5 Gabriel Infante-Lopez (UvA) *Two-Level Probabilistic Grammars for Natural Language Parsing*
6 Pieter Spronck (UM) *Adaptive Game AI*
7 Flavius Frasincar (TUe) *Hypermedia Presentation Generation for Semantic Web Information Systems*
8 Richard Vdovjak (TUe) *A Model-driven Approach for Building Distributed Ontology-based Web Applications*
9 Jeen Broekstra (VUA) *Storage, Querying and Inferencing for Semantic Web Languages*
10 Anders Bouwer (UvA) *Explaining Behaviour: Using Qualitative Simulation in Interactive Learning Environments*
11 Elth Ogston (VUA) *Agent Based Matchmaking and Clustering: A Decentralized Approach to Search*
12 Csaba Boer (EUR) *Distributed Simulation in Industry*
13 Fred Hamburg (UL) *Een Computermodel voor het Ondersteunen van Euthanasiebeslissingen*
14 Borys Omelayenko (VUA) *Web-Service configuration on the Semantic Web: Exploring how semantics meets pragmatics*
15 Tibor Bosse (VUA) *Analysis of the Dynamics of Cognitive Processes*
16 Joris Graaumans (UU) *Usability of XML Query Languages*
17 Boris Shishkov (TUD) *Software Specification Based on Re-usable Business Components*
18 Danielle Sent (UU) *Test-selection strategies for probabilistic networks*
19 Michel van Dartel (UM) *Situated Representation*
20 Cristina Coteanu (UL) *Cyber Consumer Law, State of the Art and Perspectives*
21 Wijnand Derks (UT) *Improving Concurrency and Recovery in Database Systems by Exploiting Application Semantics*

2006

1 Samuil Angelov (TUe) *Foundations of B2B Electronic Contracting*
2 Cristina Chisalita (VUA) *Contextual issues in the design and use of information technology in organizations*
3 Noor Christoph (UvA) *The role of metacognitive skills in learning to solve problems*
4 Marta Sabou (VUA) *Building Web Service Ontologies*
5 Cees Pierik (UU) *Validation Techniques for Object-Oriented Proof Outlines*
6 Ziv Baida (VUA) *Software-aided Service Bundling: Intelligent Methods & Tools for Graphical Service Modeling*
7 Marko Smiljanic (UT) *XML schema matching: balancing efficiency and effectiveness by means of clustering*
8 Eelco Herder (UT) *Forward, Back and Home Again: Analyzing User Behavior on the Web*
9 Mohamed Wahdan (UM) *Automatic Formulation of the Auditor's Opinion*
10 Ronny Siebes (VUA) *Semantic Routing in Peer-to-Peer Systems*
11 Joeri van Ruth (UT) *Flattening Queries over Nested Data Types*
12 Bert Bongers (VUA) *Interactivation: Towards an e-cology of people, our technological environment, and the arts*
13 Henk-Jan Lebbink (UU) *Dialogue and Decision Games for Information Exchanging Agents*
14 Johan Hoorn (VUA) *Software Requirements: Update, Upgrade, Redesign - towards a Theory of Requirements Change*
15 Rainer Malik (UU) *CONAN: Text Mining in the Biomedical Domain*
16 Carsten Riggelsen (UU) *Approximation Methods for Efficient Learning of Bayesian Networks*

17 Stacey Nagata (UU) *User Assistance for Multitasking with Interruptions on a Mobile Device*
18 Valentin Zhizhkun (UvA) *Graph transformation for Natural Language Processing*
19 Birna van Riemsdijk (UU) *Cognitive Agent Programming: A Semantic Approach*
20 Marina Velikova (UvT) *Monotone models for prediction in data mining*
21 Bas van Gils (RUN) *Aptness on the Web*
22 Paul de Vrieze (RUN) *Fundaments of Adaptive Personalisation*
23 Ion Juvina (UU) *Development of Cognitive Model for Navigating on the Web*
24 Laura Hollink (VUA) *Semantic Annotation for Retrieval of Visual Resources*
25 Madalina Drugan (UU) *Conditional log-likelihood MDL and Evolutionary MCMC*
26 Vojkan Mihajlovic (UT) *Score Region Algebra: A Flexible Framework for Structured Information Retrieval*
27 Stefano Bocconi (CWI) *Vox Populi: generating video documentaries from semantically annotated media repositories*
28 Borkur Sigurbjornsson (UvA) *Focused Information Access using XML Element Retrieval*

2007

1 Kees Leune (UvT) *Access Control and Service-Oriented Architectures*
2 Wouter Teepe (RUG) *Reconciling Information Exchange and Confidentiality: A Formal Approach*
3 Peter Mika (VUA) *Social Networks and the Semantic Web*
4 Jurriaan van Diggelen (UU) *Achieving Semantic Interoperability in Multi-agent Systems: a dialogue-based approach*
5 Bart Schermer (UL) *Software Agents, Surveillance, and the Right to Privacy: a Legislative Framework for Agent-enabled Surveillance*
6 Gilad Mishne (UvA) *Applied Text Analytics for Blogs*
7 Natasa Jovanovic' (UT) *To Whom It May Concern: Addressee Identification in Face-to-Face Meetings*
8 Mark Hoogendoorn (VUA) *Modeling of Change in Multi-Agent Organizations*
9 David Mobach (VUA) *Agent-Based Mediated Service Negotiation*
10 Huib Aldewereld (UU) *Autonomy vs. Conformity: an Institutional Perspective on Norms and Protocols*
11 Natalia Stash (TUe) *Incorporating Cognitive/Learning Styles in a General-Purpose Adaptive Hypermedia System*
12 Marcel van Gerven (RUN) *Bayesian Networks for Clinical Decision Support: A Rational Approach to Dynamic Decision-Making under Uncertainty*
13 Rutger Rienks (UT) *Meetings in Smart Environments: Implications of Progressing Technology*
14 Niek Bergboer (UM) *Context-Based Image Analysis*
15 Joyca Lacroix (UM) *NIM: a Situated Computational Memory Model*
16 Davide Grossi (UU) *Designing Invisible Handcuffs. Formal investigations in Institutions and Organizations for Multi-agent Systems*
17 Theodore Charitos (UU) *Reasoning with Dynamic Networks in Practice*
18 Bart Orriens (UvT) *On the development an management of adaptive business collaborations*
19 David Levy (UM) *Intimate relationships with artificial partners*
20 Slinger Jansen (UU) *Customer Configuration Updating in a Software Supply Network*
21 Karianne Vermaas (UU) *Fast diffusion and broadening use: A research on residential adoption and usage of broadband internet in the Netherlands between 2001 and 2005*
22 Zlatko Zlatev (UT) *Goal-oriented design of value and process models from patterns*
23 Peter Barna (TUe) *Specification of Application Logic in Web Information Systems*
24 Georgina Ramírez Camps (CWI) *Structural Features in XML Retrieval*
25 Joost Schalken (VUA) *Empirical Investigations in Software Process Improvement*

2008

1 Katalin Boer-Sorbán (EUR) *Agent-Based Simulation of Financial Markets: A modular, continuous-time approach*
2 Alexei Sharpanskykh (VUA) *On Computer-Aided Methods for Modeling and Analysis of Organizations*
3 Vera Hollink (UvA) *Optimizing hierarchical menus: a usage-based approach*
4 Ander de Keijzer (UT) *Management of Uncertain Data: towards unattended integration*
5 Bela Mutschler (UT) *Modeling and simulating causal dependencies on process-aware information systems from a cost perspective*
6 Arjen Hommersom (RUN) *On the Application of Formal Methods to Clinical Guidelines, an Artificial Intelligence Perspective*
7 Peter van Rosmalen (OU) *Supporting the tutor in the design and support of adaptive e-learning*
8 Janneke Bolt (UU) *Bayesian Networks: Aspects of Approximate Inference*
9 Christof van Nimwegen (UU) *The paradox of the guided user: assistance can be counter-effective*

10 Wauter Bosma (UT) *Discourse oriented summarization*
11 Vera Kartseva (VUA) *Designing Controls for Network Organizations: A Value-Based Approach*
12 Jozsef Farkas (RUN) *A Semiotically Oriented Cognitive Model of Knowledge Representation*
13 Caterina Carraciolo (UvA) *Topic Driven Access to Scientific Handbooks*
14 Arthur van Bunningen (UT) *Context-Aware Querying: Better Answers with Less Effort*
15 Martijn van Otterlo (UT) *The Logic of Adaptive Behavior: Knowledge Representation and Algorithms for the Markov Decision Process Framework in First-Order Domains*
16 Henriette van Vugt (VUA) *Embodied agents from a user's perspective*
17 Martin Op 't Land (TUD) *Applying Architecture and Ontology to the Splitting and Allying of Enterprises*
18 Guido de Croon (UM) *Adaptive Active Vision*
19 Henning Rode (UT) *From Document to Entity Retrieval: Improving Precision and Performance of Focused Text Search*
20 Rex Arendsen (UvA) *Geen bericht, goed bericht. Een onderzoek naar de effecten van de introductie van elektronisch berichtenverkeer met de overheid op de administratieve lasten van bedrijven*
21 Krisztian Balog (UvA) *People Search in the Enterprise*
22 Henk Koning (UU) *Communication of IT-Architecture*
23 Stefan Visscher (UU) *Bayesian network models for the management of ventilator-associated pneumonia*
24 Zharko Aleksovski (VUA) *Using background knowledge in ontology matching*
25 Geert Jonker (UU) *Efficient and Equitable Exchange in Air Traffic Management Plan Repair using Spender-signed Currency*
26 Marijn Huijbregts (UT) *Segmentation, Diarization and Speech Transcription: Surprise Data Unraveled*
27 Hubert Vogten (OU) *Design and Implementation Strategies for IMS Learning Design*
28 Ildiko Flesch (RUN) *On the Use of Independence Relations in Bayesian Networks*
29 Dennis Reidsma (UT) *Annotations and Subjective Machines: Of Annotators, Embodied Agents, Users, and Other Humans*
30 Wouter van Atteveldt (VUA) *Semantic Network Analysis: Techniques for Extracting, Representing and Querying Media Content*
31 Loes Braun (UM) *Pro-Active Medical Information Retrieval*
32 Trung H. Bui (UT) *Toward Affective Dialogue Management using Partially Observable Markov Decision Processes*
33 Frank Terpstra (UvA) *Scientific Workflow Design: theoretical and practical issues*
34 Jeroen de Knijf (UU) *Studies in Frequent Tree Mining*
35 Ben Torben Nielsen (UvT) *Dendritic morphologies: function shapes structure*

2009

1 Rasa Jurgelenaite (RUN) *Symmetric Causal Independence Models*
2 Willem Robert van Hage (VUA) *Evaluating Ontology-Alignment Techniques*
3 Hans Stol (UvT) *A Framework for Evidence-based Policy Making Using IT*
4 Josephine Nabukenya (RUN) *Improving the Quality of Organisational Policy Making using Collaboration Engineering*
5 Sietse Overbeek (RUN) *Bridging Supply and Demand for Knowledge Intensive Tasks: Based on Knowledge, Cognition, and Quality*
6 Muhammad Subianto (UU) *Understanding Classification*
7 Ronald Poppe (UT) *Discriminative Vision-Based Recovery and Recognition of Human Motion*
8 Volker Nannen (VUA) *Evolutionary Agent-Based Policy Analysis in Dynamic Environments*
9 Benjamin Kanagwa (RUN) *Design, Discovery and Construction of Service-oriented Systems*
10 Jan Wielemaker (UvA) *Logic programming for knowledge-intensive interactive applications*
11 Alexander Boer (UvA) *Legal Theory, Sources of Law & the Semantic Web*
12 Peter Massuthe (TUE, Humboldt-Universitaet zu Berlin) *Operating Guidelines for Services*
13 Steven de Jong (UM) *Fairness in Multi-Agent Systems*
14 Maksym Korotkiy (VUA) *From ontology-enabled services to service-enabled ontologies (making ontologies work in e-science with ONTO-SOA)*
15 Rinke Hoekstra (UvA) *Ontology Representation: Design Patterns and Ontologies that Make Sense*
16 Fritz Reul (UvT) *New Architectures in Computer Chess*
17 Laurens van der Maaten (UvT) *Feature Extraction from Visual Data*
18 Fabian Groffen (CWI) *Armada, An Evolving Database System*
19 Valentin Robu (CWI) *Modeling Preferences, Strategic Reasoning and Collaboration in Agent-Mediated Electronic Markets*
20 Bob van der Vecht (UU) *Adjustable Autonomy: Controling Influences on Decision Making*
21 Stijn Vanderlooy (UM) *Ranking and Reliable Classification*
22 Pavel Serdyukov (UT) *Search For Expertise: Going beyond direct evidence*

23 Peter Hofgesang (VUA) *Modelling Web Usage in a Changing Environment*
24 Annerieke Heuvelink (VUA) *Cognitive Models for Training Simulations*
25 Alex van Ballegooij (CWI) *RAM: Array Database Management through Relational Mapping*
26 Fernando Koch (UU) *An Agent-Based Model for the Development of Intelligent Mobile Services*
27 Christian Glahn (OU) *Contextual Support of social Engagement and Reflection on the Web*
28 Sander Evers (UT) *Sensor Data Management with Probabilistic Models*
29 Stanislav Pokraev (UT) *Model-Driven Semantic Integration of Service-Oriented Applications*
30 Marcin Zukowski (CWI) *Balancing vectorized query execution with bandwidth-optimized storage*
31 Sofiya Katrenko (UvA) *A Closer Look at Learning Relations from Text*
32 Rik Farenhorst (VUA) *Architectural Knowledge Management: Supporting Architects and Auditors*
33 Khiet Truong (UT) *How Does Real Affect Affect Affect Recognition In Speech?*
34 Inge van de Weerd (UU) *Advancing in Software Product Management: An Incremental Method Engineering Approach*
35 Wouter Koelewijn (UL) *Privacy en Politiegegevens: Over geautomatiseerde normatieve informatie-uitwisseling*
36 Marco Kalz (OUN) *Placement Support for Learners in Learning Networks*
37 Hendrik Drachsler (OUN) *Navigation Support for Learners in Informal Learning Networks*
38 Riina Vuorikari (OU) *Tags and self-organisation: a metadata ecology for learning resources in a multilingual context*
39 Christian Stahl (TUE, Humboldt-Universitaet zu Berlin) *Service Substitution: A Behavioral Approach Based on Petri Nets*
40 Stephan Raaijmakers (UvT) *Multinomial Language Learning: Investigations into the Geometry of Language*
41 Igor Berezhnyy (UvT) *Digital Analysis of Paintings*
42 Toine Bogers (UvT) *Recommender Systems for Social Bookmarking*
43 Virginia Nunes Leal Franqueira (UT) *Finding Multi-step Attacks in Computer Networks using Heuristic Search and Mobile Ambients*
44 Roberto Santana Tapia (UT) *Assessing Business-IT Alignment in Networked Organizations*
45 Jilles Vreeken (UU) *Making Pattern Mining Useful*
46 Loredana Afanasiev (UvA) *Querying XML: Benchmarks and Recursion*

2010

1 Matthijs van Leeuwen (UU) *Patterns that Matter*
2 Ingo Wassink (UT) *Work flows in Life Science*
3 Joost Geurts (CWI) *A Document Engineering Model and Processing Framework for Multimedia documents*
4 Olga Kulyk (UT) *Do You Know What I Know? Situational Awareness of Co-located Teams in Multidisplay Environments*
5 Claudia Hauff (UT) *Predicting the Effectiveness of Queries and Retrieval Systems*
6 Sander Bakkes (UvT) *Rapid Adaptation of Video Game AI*
7 Wim Fikkert (UT) *Gesture interaction at a Distance*
8 Krzysztof Siewicz (UL) *Towards an Improved Regulatory Framework of Free Software. Protecting user freedoms in a world of software communities and eGovernments*
9 Hugo Kielman (UL) *A Politiele gegevensverwerking en Privacy, Naar een effectieve waarborging*
10 Rebecca Ong (UL) *Mobile Communication and Protection of Children*
11 Adriaan Ter Mors (TUD) *The world according to MARP: Multi-Agent Route Planning*
12 Susan van den Braak (UU) *Sensemaking software for crime analysis*
13 Gianluigi Folino (RUN) *High Performance Data Mining using Bio-inspired techniques*
14 Sander van Splunter (VUA) *Automated Web Service Reconfiguration*
15 Lianne Bodenstaff (UT) *Managing Dependency Relations in Inter-Organizational Models*
16 Sicco Verwer (TUD) *Efficient Identification of Timed Automata, theory and practice*
17 Spyros Kotoulas (VUA) *Scalable Discovery of Networked Resources: Algorithms, Infrastructure, Applications*
18 Charlotte Gerritsen (VUA) *Caught in the Act: Investigating Crime by Agent-Based Simulation*
19 Henriette Cramer (UvA) *People's Responses to Autonomous and Adaptive Systems*
20 Ivo Swartjes (UT) *Whose Story Is It Anyway? How Improv Informs Agency and Authorship of Emergent Narrative*
21 Harold van Heerde (UT) *Privacy-aware data management by means of data degradation*
22 Michiel Hildebrand (CWI) *End-user Support for Access to Heterogeneous Linked Data*
23 Bas Steunebrink (UU) *The Logical Structure of Emotions*
24 Zulfiqar Ali Memon (VUA) *Modelling Human-Awareness for Ambient Agents: A Human Mindreading Perspective*
25 Ying Zhang (CWI) *XRPC: Efficient Distributed Query Processing on Heterogeneous XQuery Engines*
26 Marten Voulon (UL) *Automatisch contracteren*

27 Arne Koopman (UU) *Characteristic Relational Patterns*
28 Stratos Idreos (CWI) *Database Cracking: Towards Auto-tuning Database Kernels*
29 Marieke van Erp (UvT) *Accessing Natural History: Discoveries in data cleaning, structuring, and retrieval*
30 Victor de Boer (UvA) *Ontology Enrichment from Heterogeneous Sources on the Web*
31 Marcel Hiel (UvT) *An Adaptive Service Oriented Architecture: Automatically solving Interoperability Problems*
32 Robin Aly (UT) *Modeling Representation Uncertainty in Concept-Based Multimedia Retrieval*
33 Teduh Dirgahayu (UT) *Interaction Design in Service Compositions*
34 Dolf Trieschnigg (UT) *Proof of Concept: Concept-based Biomedical Information Retrieval*
35 Jose Janssen (OU) *Paving the Way for Lifelong Learning: Facilitating competence development through a learning path specification*
36 Niels Lohmann (TUe) *Correctness of services and their composition*
37 Dirk Fahland (TUe) *From Scenarios to components*
38 Ghazanfar Farooq Siddiqui (VUA) *Integrative modeling of emotions in virtual agents*
39 Mark van Assem (VUA) *Converting and Integrating Vocabularies for the Semantic Web*
40 Guillaume Chaslot (UM) *Monte-Carlo Tree Search*
41 Sybren de Kinderen (VUA) *Needs-driven service bundling in a multi-supplier setting: the computational e3-service approach*
42 Peter van Kranenburg (UU) *A Computational Approach to Content-Based Retrieval of Folk Song Melodies*
43 Pieter Bellekens (TUe) *An Approach towards Context-sensitive and User-adapted Access to Heterogeneous Data Sources, Illustrated in the Television Domain*
44 Vasilios Andrikopoulos (UvT) *A theory and model for the evolution of software services*
45 Vincent Pijpers (VUA) *e3alignment: Exploring Inter-Organizational Business-ICT Alignment*
46 Chen Li (UT) *Mining Process Model Variants: Challenges, Techniques, Examples*
47 Jahn-Takeshi Saito (UM) *Solving difficult game positions*
48 Bouke Huurnink (UvA) *Search in Audiovisual Broadcast Archives*
49 Alia Khairia Amin (CWI) *Understanding and supporting information seeking tasks in multiple sources*
50 Peter-Paul van Maanen (VUA) *Adaptive Support for Human-Computer Teams: Exploring the Use of Cognitive Models of Trust and Attention*
51 Edgar Meij (UvA) *Combining Concepts and Language Models for Information Access*

2011

1 Botond Cseke (RUN) *Variational Algorithms for Bayesian Inference in Latent Gaussian Models*
2 Nick Tinnemeier (UU) *Organizing Agent Organizations. Syntax and Operational Semantics of an Organization-Oriented Programming Language*
3 Jan Martijn van der Werf (TUe) *Compositional Design and Verification of Component-Based Information Systems*
4 Hado van Hasselt (UU) *Insights in Reinforcement Learning: Formal analysis and empirical evaluation of temporal-difference*
5 Base van der Raadt (VUA) *Enterprise Architecture Coming of Age: Increasing the Performance of an Emerging Discipline*
6 Yiwen Wang (TUe) *Semantically-Enhanced Recommendations in Cultural Heritage*
7 Yujia Cao (UT) *Multimodal Information Presentation for High Load Human Computer Interaction*
8 Nieske Vergunst (UU) *BDI-based Generation of Robust Task-Oriented Dialogues*
9 Tim de Jong (OU) *Contextualised Mobile Media for Learning*
10 Bart Bogaert (UvT) *Cloud Content Contention*
11 Dhaval Vyas (UT) *Designing for Awareness: An Experience-focused HCI Perspective*
12 Carmen Bratosin (TUe) *Grid Architecture for Distributed Process Mining*
13 Xiaoyu Mao (UvT) *Airport under Control. Multi-agent Scheduling for Airport Ground Handling*
14 Milan Lovric (EUR) *Behavioral Finance and Agent-Based Artificial Markets*
15 Marijn Koolen (UvA) *The Meaning of Structure: the Value of Link Evidence for Information Retrieval*
16 Maarten Schadd (UM) *Selective Search in Games of Different Complexity*
17 Jiyin He (UvA) *Exploring Topic Structure: Coherence, Diversity and Relatedness*
18 Mark Ponsen (UM) *Strategic Decision-Making in complex games*
19 Ellen Rusman (OU) *The Mind ' s Eye on Personal Profiles*
20 Qing Gu (VUA) *Guiding service-oriented software engineering: A view-based approach*
21 Linda Terlouw (TUD) *Modularization and Specification of Service-Oriented Systems*
22 Junte Zhang (UvA) *System Evaluation of Archival Description and Access*
23 Wouter Weerkamp (UvA) *Finding People and their Utterances in Social Media*

24 Herwin van Welbergen (UT) *Behavior Generation for Interpersonal Coordination with Virtual Humans On Specifying, Scheduling and Realizing Multimodal Virtual Human Behavior*
25 Syed Waqar ul Qounain Jaffry (VUA) *Analysis and Validation of Models for Trust Dynamics*
26 Matthijs Aart Pontier (VUA) *Virtual Agents for Human Communication: Emotion Regulation and Involvement-Distance Trade-Offs in Embodied Conversational Agents and Robots*
27 Aniel Bhulai (VUA) *Dynamic website optimization through autonomous management of design patterns*
28 Rianne Kaptein (UvA) *Effective Focused Retrieval by Exploiting Query Context and Document Structure*
29 Faisal Kamiran (TUe) *Discrimination-aware Classification*
30 Egon van den Broek (UT) *Affective Signal Processing (ASP): Unraveling the mystery of emotions*
31 Ludo Waltman (EUR) *Computational and Game-Theoretic Approaches for Modeling Bounded Rationality*
32 Nees-Jan van Eck (EUR) *Methodological Advances in Bibliometric Mapping of Science*
33 Tom van der Weide (UU) *Arguing to Motivate Decisions*
34 Paolo Turrini (UU) *Strategic Reasoning in Interdependence: Logical and Game-theoretical Investigations*
35 Maaike Harbers (UU) *Explaining Agent Behavior in Virtual Training*
36 Erik van der Spek (UU) *Experiments in serious game design: a cognitive approach*
37 Adriana Burlutiu (RUN) *Machine Learning for Pairwise Data, Applications for Preference Learning and Supervised Network Inference*
38 Nyree Lemmens (UM) *Bee-inspired Distributed Optimization*
39 Joost Westra (UU) *Organizing Adaptation using Agents in Serious Games*
40 Viktor Clerc (VUA) *Architectural Knowledge Management in Global Software Development*
41 Luan Ibraimi (UT) *Cryptographically Enforced Distributed Data Access Control*
42 Michal Sindlar (UU) *Explaining Behavior through Mental State Attribution*
43 Henk van der Schuur (UU) *Process Improvement through Software Operation Knowledge*
44 Boris Reuderink (UT) *Robust Brain-Computer Interfaces*
45 Herman Stehouwer (UvT) *Statistical Language Models for Alternative Sequence Selection*
46 Beibei Hu (TUD) *Towards Contextualized Information Delivery: A Rule-based Architecture for the Domain of Mobile Police Work*
47 Azizi Bin Ab Aziz (VUA) *Exploring Computational Models for Intelligent Support of Persons with Depression*
48 Mark Ter Maat (UT) *Response Selection and Turn-taking for a Sensitive Artificial Listening Agent*
49 Andreea Niculescu (UT) *Conversational interfaces for task-oriented spoken dialogues: design aspects influencing interaction quality*

2012

1 Terry Kakeeto (UvT) *Relationship Marketing for SMEs in Uganda*
2 Muhammad Umair (VUA) *Adaptivity, emotion, and Rationality in Human and Ambient Agent Models*
3 Adam Vanya (VUA) *Supporting Architecture Evolution by Mining Software Repositories*
4 Jurriaan Souer (UU) *Development of Content Management System-based Web Applications*
5 Marijn Plomp (UU) *Maturing Interorganisational Information Systems*
6 Wolfgang Reinhardt (OU) *Awareness Support for Knowledge Workers in Research Networks*
7 Rianne van Lambalgen (VUA) *When the Going Gets Tough: Exploring Agent-based Models of Human Performance under Demanding Conditions*
8 Gerben de Vries (UvA) *Kernel Methods for Vessel Trajectories*
9 Ricardo Neisse (UT) *Trust and Privacy Management Support for Context-Aware Service Platforms*
10 David Smits (TUe) *Towards a Generic Distributed Adaptive Hypermedia Environment*
11 J. C. B. Rantham Prabhakara (TUe) *Process Mining in the Large: Preprocessing, Discovery, and Diagnostics*
12 Kees van der Sluijs (TUe) *Model Driven Design and Data Integration in Semantic Web Information Systems*
13 Suleman Shahid (UvT) *Fun and Face: Exploring non-verbal expressions of emotion during playful interactions*
14 Evgeny Knutov (TUe) *Generic Adaptation Framework for Unifying Adaptive Web-based Systems*
15 Natalie van der Wal (VUA) *Social Agents. Agent-Based Modelling of Integrated Internal and Social Dynamics of Cognitive and Affective Processes*
16 Fiemke Both (VUA) *Helping people by understanding them: Ambient Agents supporting task execution and depression treatment*
17 Amal Elgammal (UvT) *Towards a Comprehensive Framework for Business Process Compliance*
18 Eltjo Poort (VUA) *Improving Solution Architecting Practices*
19 Helen Schonenberg (TUe) *What's Next? Operational Support for Business Process Execution*

20 Ali Bahramisharif (RUN) *Covert Visual Spatial Attention, a Robust Paradigm for Brain-Computer Interfacing*
21 Roberto Cornacchia (TUD) *Querying Sparse Matrices for Information Retrieval*
22 Thijs Vis (UvT) *Intelligence, politie en veiligheidsdienst: verenigbare grootheden?*
23 Christian Muehl (UT) *Toward Affective Brain-Computer Interfaces: Exploring the Neurophysiology of Affect during Human Media Interaction*
24 Laurens van der Werff (UT) *Evaluation of Noisy Transcripts for Spoken Document Retrieval*
25 Silja Eckartz (UT) *Managing the Business Case Development in Inter-Organizational IT Projects: A Methodology and its Application*
26 Emile de Maat (UvA) *Making Sense of Legal Text*
27 Hayrettin Gurkok (UT) *Mind the Sheep! User Experience Evaluation & Brain-Computer Interface Games*
28 Nancy Pascall (UvT) *Engendering Technology Empowering Women*
29 Almer Tigelaar (UT) *Peer-to-Peer Information Retrieval*
30 Alina Pommeranz (TUD) *Designing Human-Centered Systems for Reflective Decision Making*
31 Emily Bagarukayo (RUN) *A Learning by Construction Approach for Higher Order Cognitive Skills Improvement, Building Capacity and Infrastructure*
32 Wietske Visser (TUD) *Qualitative multi-criteria preference representation and reasoning*
33 Rory Sie (OUN) *Coalitions in Cooperation Networks (COCOON)*
34 Pavol Jancura (RUN) *Evolutionary analysis in PPI networks and applications*
35 Evert Haasdijk (VUA) *Never Too Old To Learn: On-line Evolution of Controllers in Swarm- and Modular Robotics*
36 Denis Ssebugwawo (RUN) *Analysis and Evaluation of Collaborative Modeling Processes*
37 Agnes Nakakawa (RUN) *A Collaboration Process for Enterprise Architecture Creation*
38 Selmar Smit (VUA) *Parameter Tuning and Scientific Testing in Evolutionary Algorithms*
39 Hassan Fatemi (UT) *Risk-aware design of value and coordination networks*
40 Agus Gunawan (UvT) *Information Access for SMEs in Indonesia*
41 Sebastian Kelle (OU) *Game Design Patterns for Learning*
42 Dominique Verpoorten (OU) *Reflection Amplifiers in self-regulated Learning*
43 Anna Tordai (VUA) *On Combining Alignment Techniques*
44 Benedikt Kratz (UvT) *A Model and Language for Business-aware Transactions*
45 Simon Carter (UvA) *Exploration and Exploitation of Multilingual Data for Statistical Machine Translation*
46 Manos Tsagkias (UvA) *Mining Social Media: Tracking Content and Predicting Behavior*
47 Jorn Bakker (TUe) *Handling Abrupt Changes in Evolving Time-series Data*
48 Michael Kaisers (UM) *Learning against Learning: Evolutionary dynamics of reinforcement learning algorithms in strategic interactions*
49 Steven van Kervel (TUD) *Ontologogy driven Enterprise Information Systems Engineering*
50 Jeroen de Jong (TUD) *Heuristics in Dynamic Sceduling: a practical framework with a case study in elevator dispatching*

2013

1 Viorel Milea (EUR) *News Analytics for Financial Decision Support*
2 Erietta Liarou (CWI) *MonetDB/DataCell: Leveraging the Column-store Database Technology for Efficient and Scalable Stream Processing*
3 Szymon Klarman (VUA) *Reasoning with Contexts in Description Logics*
4 Chetan Yadati (TUD) *Coordinating autonomous planning and scheduling*
5 Dulce Pumareja (UT) *Groupware Requirements Evolutions Patterns*
6 Romulo Goncalves (CWI) *The Data Cyclotron: Juggling Data and Queries for a Data Warehouse Audience*
7 Giel van Lankveld (UvT) *Quantifying Individual Player Differences*
8 Robbert-Jan Merk (VUA) *Making enemies: cognitive modeling for opponent agents in fighter pilot simulators*
9 Fabio Gori (RUN) *Metagenomic Data Analysis: Computational Methods and Applications*
10 Jeewanie Jayasinghe Arachchige (UvT) *A Unified Modeling Framework for Service Design*
11 Evangelos Pournaras (TUD) *Multi-level Reconfigurable Self-organization in Overlay Services*
12 Marian Razavian (VUA) *Knowledge-driven Migration to Services*
13 Mohammad Safiri (UT) *Service Tailoring: User-centric creation of integrated IT-based homecare services to support independent living of elderly*
14 Jafar Tanha (UvA) *Ensemble Approaches to Semi-Supervised Learning Learning*
15 Daniel Hennes (UM) *Multiagent Learning: Dynamic Games and Applications*
16 Eric Kok (UU) *Exploring the practical benefits of argumentation in multi-agent deliberation*
17 Koen Kok (VUA) *The PowerMatcher: Smart Coordination for the Smart Electricity Grid*
18 Jeroen Janssens (UvT) *Outlier Selection and One-Class Classification*

19 Renze Steenhuizen (TUD) *Coordinated Multi-Agent Planning and Scheduling*
20 Katja Hofmann (UvA) *Fast and Reliable Online Learning to Rank for Information Retrieval*
21 Sander Wubben (UvT) *Text-to-text generation by monolingual machine translation*
22 Tom Claassen (RUN) *Causal Discovery and Logic*
23 Patricio de Alencar Silva (UvT) *Value Activity Monitoring*
24 Haitham Bou Ammar (UM) *Automated Transfer in Reinforcement Learning*
25 Agnieszka Anna Latoszek-Berendsen (UM) *Intention-based Decision Support. A new way of representing and implementing clinical guidelines in a Decision Support System*
26 Alireza Zarghami (UT) *Architectural Support for Dynamic Homecare Service Provisioning*
27 Mohammad Huq (UT) *Inference-based Framework Managing Data Provenance*
28 Frans van der Sluis (UT) *When Complexity becomes Interesting: An Inquiry into the Information eXperience*
29 Iwan de Kok (UT) *Listening Heads*
30 Joyce Nakatumba (TUe) *Resource-Aware Business Process Management: Analysis and Support*
31 Dinh Khoa Nguyen (UvT) *Blueprint Model and Language for Engineering Cloud Applications*
32 Kamakshi Rajagopal (OUN) *Networking For Learning: The role of Networking in a Lifelong Learner's Professional Development*
33 Qi Gao (TUD) *User Modeling and Personalization in the Microblogging Sphere*
34 Kien Tjin-Kam-Jet (UT) *Distributed Deep Web Search*
35 Abdallah El Ali (UvA) *Minimal Mobile Human Computer Interaction*
36 Than Lam Hoang (TUe) *Pattern Mining in Data Streams*
37 Dirk Börner (OUN) *Ambient Learning Displays*
38 Eelco den Heijer (VUA) *Autonomous Evolutionary Art*
39 Joop de Jong (TUD) *A Method for Enterprise Ontology based Design of Enterprise Information Systems*
40 Pim Nijssen (UM) *Monte-Carlo Tree Search for Multi-Player Games*
41 Jochem Liem (UvA) *Supporting the Conceptual Modelling of Dynamic Systems: A Knowledge Engineering Perspective on Qualitative Reasoning*
42 Léon Planken (TUD) *Algorithms for Simple Temporal Reasoning*
43 Marc Bron (UvA) *Exploration and Contextualization through Interaction and Concepts*

2014

1 Nicola Barile (UU) *Studies in Learning Monotone Models from Data*
2 Fiona Tuliyano (RUN) *Combining System Dynamics with a Domain Modeling Method*
3 Sergio Raul Duarte Torres (UT) *Information Retrieval for Children: Search Behavior and Solutions*
4 Hanna Jochmann-Mannak (UT) *Websites for children: search strategies and interface design - Three studies on children's search performance and evaluation*
5 Jurriaan van Reijsen (UU) *Knowledge Perspectives on Advancing Dynamic Capability*
6 Damian Tamburri (VUA) *Supporting Networked Software Development*
7 Arya Adriansyah (TUe) *Aligning Observed and Modeled Behavior*
8 Samur Araujo (TUD) *Data Integration over Distributed and Heterogeneous Data Endpoints*
9 Philip Jackson (UvT) *Toward Human-Level Artificial Intelligence: Representation and Computation of Meaning in Natural Language*
10 Ivan Salvador Razo Zapata (VUA) *Service Value Networks*
11 Janneke van der Zwaan (TUD) *An Empathic Virtual Buddy for Social Support*
12 Willem van Willigen (VUA) *Look Ma, No Hands: Aspects of Autonomous Vehicle Control*
13 Arlette van Wissen (VUA) *Agent-Based Support for Behavior Change: Models and Applications in Health and Safety Domains*
14 Yangyang Shi (TUD) *Language Models With Meta-information*
15 Natalya Mogles (VUA) *Agent-Based Analysis and Support of Human Functioning in Complex Socio-Technical Systems: Applications in Safety and Healthcare*
16 Krystyna Milian (VUA) *Supporting trial recruitment and design by automatically interpreting eligibility criteria*
17 Kathrin Dentler (VUA) *Computing healthcare quality indicators automatically: Secondary Use of Patient Data and Semantic Interoperability*
18 Mattijs Ghijsen (UvA) *Methods and Models for the Design and Study of Dynamic Agent Organizations*
19 Vinicius Ramos (TUe) *Adaptive Hypermedia Courses: Qualitative and Quantitative Evaluation and Tool Support*
20 Mena Habib (UT) *Named Entity Extraction and Disambiguation for Informal Text: The Missing Link*
21 Kassidy Clark (TUD) *Negotiation and Monitoring in Open Environments*
22 Marieke Peeters (UU) *Personalized Educational Games: Developing agent-supported scenario-based training*

23 Eleftherios Sidirourgos (UvA/CWI) *Space Efficient Indexes for the Big Data Era*
24 Davide Ceolin (VUA) *Trusting Semi-structured Web Data*
25 Martijn Lappenschaar (RUN) *New network models for the analysis of disease interaction*
26 Tim Baarslag (TUD) *What to Bid and When to Stop*
27 Rui Jorge Almeida (EUR) *Conditional Density Models Integrating Fuzzy and Probabilistic Representations of Uncertainty*
28 Anna Chmielowiec (VUA) *Decentralized k-Clique Matching*
29 Jaap Kabbedijk (UU) *Variability in Multi-Tenant Enterprise Software*
30 Peter de Cock (UvT) *Anticipating Criminal Behaviour*
31 Leo van Moergestel (UU) *Agent Technology in Agile Multiparallel Manufacturing and Product Support*
32 Naser Ayat (UvA) *On Entity Resolution in Probabilistic Data*
33 Tesfa Tegegne (RUN) *Service Discovery in eHealth*
34 Christina Manteli (VUA) *The Effect of Governance in Global Software Development: Analyzing Transactive Memory Systems*
35 Joost van Ooijen (UU) *Cognitive Agents in Virtual Worlds: A Middleware Design Approach*
36 Joos Buijs (TUe) *Flexible Evolutionary Algorithms for Mining Structured Process Models*
37 Maral Dadvar (UT) *Experts and Machines United Against Cyberbullying*
38 Danny Plass-Oude Bos (UT) *Making brain-computer interfaces better: improving usability through post-processing*
39 Jasmina Maric (UvT) *Web Communities, Immigration, and Social Capital*
40 Walter Omona (RUN) *A Framework for Knowledge Management Using ICT in Higher Education*
41 Frederic Hogenboom (EUR) *Automated Detection of Financial Events in News Text*
42 Carsten Eijckhof (CWI/TUD) *Contextual Multidimensional Relevance Models*
43 Kevin Vlaanderen (UU) *Supporting Process Improvement using Method Increments*
44 Paulien Meesters (UvT) *Intelligent Blauw: Intelligence-gestuurde politiezorg in gebiedsgebonden eenheden*
45 Birgit Schmitz (OUN) *Mobile Games for Learning: A Pattern-Based Approach*
46 Ke Tao (TUD) *Social Web Data Analytics: Relevance, Redundancy, Diversity*
47 Shangsong Liang (UvA) *Fusion and Diversification in Information Retrieval*

2015

1 Niels Netten (UvA) *Machine Learning for Relevance of Information in Crisis Response*
2 Faiza Bukhsh (UvT) *Smart auditing: Innovative Compliance Checking in Customs Controls*
3 Twan van Laarhoven (RUN) *Machine learning for network data*
4 Howard Spoelstra (OUN) *Collaborations in Open Learning Environments*
5 Christoph Bösch (UT) *Cryptographically Enforced Search Pattern Hiding*
6 Farideh Heidari (TUD) *Business Process Quality Computation: Computing Non-Functional Requirements to Improve Business Processes*
7 Maria-Hendrike Peetz (UvA) *Time-Aware Online Reputation Analysis*
8 Jie Jiang (TUD) *Organizational Compliance: An agent-based model for designing and evaluating organizational interactions*
9 Randy Klaassen (UT) *HCI Perspectives on Behavior Change Support Systems*
10 Henry Hermans (OUN) *OpenU: design of an integrated system to support lifelong learning*
11 Yongming Luo (TUe) *Designing algorithms for big graph datasets: A study of computing bisimulation and joins*
12 Julie M. Birkholz (VUA) *Modi Operandi of Social Network Dynamics: The Effect of Context on Scientific Collaboration Networks*
13 Giuseppe Procaccianti (VUA) *Energy-Efficient Software*
14 Bart van Straalen (UT) *A cognitive approach to modeling bad news conversations*
15 Klaas Andries de Graaf (VUA) *Ontology-based Software Architecture Documentation*
16 Changyun Wei (UT) *Cognitive Coordination for Cooperative Multi-Robot Teamwork*
17 André van Cleeff (UT) *Physical and Digital Security Mechanisms: Properties, Combinations and Trade-offs*
18 Holger Pirk (CWI) *Waste Not, Want Not!: Managing Relational Data in Asymmetric Memories*
19 Bernardo Tabuenca (OUN) *Ubiquitous Technology for Lifelong Learners*
20 Loïs Vanhée (UU) *Using Culture and Values to Support Flexible Coordination*
21 Sibren Fetter (OUN) *Using Peer-Support to Expand and Stabilize Online Learning*
22 Zhemin Zhu (UT) *Co-occurrence Rate Networks*
23 Luit Gazendam (VUA) *Cataloguer Support in Cultural Heritage*
24 Richard Berendsen (UvA) *Finding People, Papers, and Posts: Vertical Search Algorithms and Evaluation*
25 Steven Woudenberg (UU) *Bayesian Tools for Early Disease Detection*

26 Alexander Hogenboom (EUR) *Sentiment Analysis of Text Guided by Semantics and Structure*
27 Sándor Héman (CWI) *Updating compressed colomn stores*
28 Janet Bagorogoza (TiU) *KNOWLEDGE MANAGEMENT AND HIGH PERFORMANCE: The Uganda Financial Institutions Model for HPO*
29 Hendrik Baier (UM) *Monte-Carlo Tree Search Enhancements for One-Player and Two-Player Domains*
30 Kiavash Bahreini (OU) *Real-time Multimodal Emotion Recognition in E-Learning*
31 Yakup Koç (TUD) *On the robustness of Power Grids*
32 Jerome Gard (UL) *Corporate Venture Management in SMEs*
33 Frederik Schadd (TUD) *Ontology Mapping with Auxiliary Resources*
34 Victor de Graaf (UT) *Gesocial Recommender Systems*
35 Jungxao Xu (TUD) *Affective Body Language of Humanoid Robots: Perception and Effects in Human Robot Interaction*

2016

1 Syed Saiden Abbas (RUN) *Recognition of Shapes by Humans and Machines*
2 Michiel Christiaan Meulendijk (UU) *Optimizing medication reviews through decision support: prescribing a better pill to swallow*
3 Maya Sappelli (RUN) *Knowledge Work in Context: User Centered Knowledge Worker Support*
4 Laurens Rietveld (VU) *Publishing and Consuming Linked Data*
5 Evgeny Sherkhonov (UVA) *Expanded Acyclic Queries: Containment and an Application in Explaining Missing Answers*
6 Michel Wilson (TUD) *Robust scheduling in an uncertain environment*
7 Jeroen de Man (VU) *Measuring and modeling negative emotions for virtual training*
8 Matje van de Camp (TiU) *A Link to the Past: Constructing Historical Social Networks from Unstructured Data*
9 Archana Nottamkandath (VU) *Trusting Crowdsourced Information on Cultural Artefacts*
10 George Karafotias (VUA) *Parameter Control for Evolutionary Algorithms*
11 Anne Schuth (UVA) *Search Engines that Learn from Their Users*
12 Max Knobbout (UU) *Logics for Modelling and Verifying Normative Multi-Agent Systems*
13 Nana Baah Gyan (VU) *The Web, Speech Technologies and Rural Development in West Africa - An ICT4D Approach*
14 Ravi Khadka (UU) *Revisiting Legacy Software System Modernization*
15 Steffen Michels (RUN) *Hybrid Probabilistic Logics - Theoretical Aspects, Algorithms and Experiments*
16 Guangliang Li (UVA) *Socially Intelligent Autonomous Agents that Learn from Human Reward*
17 Berend Weel (VU) *Towards Embodied Evolution of Robot Organisms*
18 Albert Meroño Peñuela (VU) *Refining Statistical Data on the Web*
19 Julia Efremova (Tu/e) *Mining Social Structures from Genealogical Data*
20 Daan Odijk (UVA) *Context & Semantics in News & Web Search*
21 Alejandro Moreno Célleri (UT) *From Traditional to Interactive Playspaces: Automatic Analysis of Player Behavior in the Interactive Tag Playground*
22 Grace Lewis (VU) *Software Architecture Strategies for Cyber-Foraging Systems*
23 Fei Cai (UVA) *Query Auto Completion in Information Retrieval*
24 Brend Wanders (UT) *Repurposing and Probabilistic Integration of Data; An Iterative and data model independent approach*
25 Julia Kiseleva (Tu/e) *Using Contextual Information to Understand Searching and Browsing Behavior*
26 Dilhan Thilakarathne (VU) *In or Out of Control: Exploring Computational Models to Study the Role of Human Awareness and Control in Behavioural Choices, with Applications in Aviation and Energy Management Domains*
27 Wen Li (TUD) *Understanding Geo-spatial Information on Social Media*
28 Mingxin Zhang (TUD) *Large-scale Agent-based Social Simulation - A study on epidemic prediction and control*
29 Nicolas Höning (TUD) *Peak reduction in decentralised electricity systems - Markets and prices for flexible planning*
30 Ruud Mattheij (UvT) *The Eyes Have It*
31 Mohammad Khelghati (UT) *Deep web content monitoring*
32 Eelco Vriezekolk (UT) *Assessing Telecommunication Service Availability Risks for Crisis Organisations*
33 Peter Bloem (UVA) *Single Sample Statistics, exercises in learning from just one example*
34 Dennis Schunselaar (TUE) *Configurable Process Trees: Elicitation, Analysis, and Enactment*
35 Zhaochun Ren (UVA) *Monitoring Social Media: Summarization, Classification and Recommendation*
36 Daphne Karreman (UT) *Beyond R2D2: The design of nonverbal interaction behavior optimized for robot-specific morphologies*

37 Giovanni Sileno (UVA) *Aligning Law and Action - a conceptual and computational inquiry*
38 Andrea Minuto (UT) *Materials that Matter - Smart Materials meet Art & Interaction Design*
39 Merijn Bruijnes (UT) *Believable Suspect Agents; Response and Interpersonal Style Selection for an Artificial Suspect*
40 Christian Detweiler (TUD) *Accounting for Values in Design*
41 Thomas King (TUD) *Governing Governance: A Formal Framework for Analysing Institutional Design and Enactment Governance*
42 Spyros Martzoukos (UVA) *Combinatorial and Compositional Aspects of Bilingual Aligned Corpora*
43 Saskia Koldijk (RUN) *Context-Aware Support for Stress Self-Management: From Theory to Practice*
44 Thibault Sellam (UVA) *Automatic Assistants for Database Exploration*
45 Bram van de Laar (UT) *Experiencing Brain-Computer Interface Control*
46 Jorge Gallego Perez (UT) *Robots to Make you Happy*
47 Christina Weber (UL) *Real-time foresight - Preparedness for dynamic innovation networks*
48 Tanja Buttler (TUD) *Collecting Lessons Learned*
49 Gleb Polevoy (TUD) *Participation and Interaction in Projects. A Game-Theoretic Analysis*
50 Yan Wang (UvT) *The Bridge of Dreams: Towards a Method for Operational Performance Alignment in IT-enabled Service Supply Chains*

2017

1 Jan-Jaap Oerlemans (UL) *Investigating Cybercrime*
2 Sjoerd Timmer (UU) *Designing and Understanding Forensic Bayesian Networks using Argumentation*
3 Daniël Harold Telgen (UU) *Grid Manufacturing; A Cyber-Physical Approach with Autonomous Products and Reconfigurable Manufacturing Machines*
4 Mrunal Gawade (CWI) *Multi-core Parallelism in a Column-store*
5 Mahdieh Shadi (UVA) *Collaboration Behavior*
6 Damir Vandic (EUR) *Intelligent Information Systems for Web Product Search*
7 Roel Bertens (UU) *Insight in Information: from Abstract to Anomaly*
8 Rob Konijn (VU) *Detecting Interesting Differences:Data Mining in Health Insurance Data using Outlier Detection and Subgroup Discovery*
9 Dong Nguyen (UT) *Text as Social and Cultural Data: A Computational Perspective on Variation in Text*
10 Robby van Delden (UT) *Steering Interactive Play Behavior*
11 Florian Kunneman (RUN) *Modelling patterns of time and emotion in Twitter #anticipointment*
12 Sander Leemans (TUE) *Robust Process Mining with Guarantees*
13 Gijs Huisman (UT) *Social Touch Technology - Extending the reach of social touch through haptic technology*
14 Shoshannah Tekofsky (UvT) *You Are Who You Play You Are: Modelling Player Traits from Video Game Behavior*
15 Peter Berck (RUN) *Memory-Based Text Correction*
16 Aleksandr Chuklin (UVA) *Understanding and Modeling Users of Modern Search Engines*
17 Daniel Dimov (UL) *Crowdsourced Online Dispute Resolution*
18 Ridho Reinanda (UVA) *Entity Associations for Search*
19 Jeroen Vuurens (UT) *Proximity of Terms, Texts and Semantic Vectors in Information Retrieval*
20 Mohammadbashir Sedighi (TUD) *Fostering Engagement in Knowledge Sharing: The Role of Perceived Benefits, Costs and Visibility*
21 Jeroen Linssen (UT) *Meta Matters in Interactive Storytelling and Serious Gaming: A Play on Worlds*
22 Sara Magliacane (VU) *Logics for causal inference under uncertainty*
23 David Graus (UVA) *Entities of Interest — Discovery in Digital Traces*
24 Chang Wang (TUD) *Use of Affordances for Efficient Robot Learning*
25 Veruska Zamborlini (VU) *Knowledge Representation for Clinical Guidelines, with applications to Multimorbidity Analysis and Literature Search*
26 Merel Jung (UT) *Socially intelligent robots that understand and respond to human touch*
27 Michiel Joosse (UT) *Investigating Positioning and Gaze Behaviors of Social Robots: People's Preferences, Perceptions and Behaviors*
28 John Klein (VU) *Architecture Practices for Complex Contexts*
29 Adel Alhuraibi (UvT) *From IT-BusinessStrategic Alignment to Performance: A Moderated Mediation Model of Social Innovation, and Enterprise Governance of IT*
30 Wilma Latuny (UvT) *The Power of Facial Expressions*
31 Ben Ruijl (UL) *Advances in computational methods for QFT calculations*
32 Thaer Samar (RUN) *Access to and Retrievability of Content in Web Archives*
33 Brigit van Loggem (OU) *Towards a Design Rationale for Software Documentation: A Model of Computer-Mediated Activity*
34 Maren Scheffel (OU) *The Evaluation Framework for Learning Analytics*

35 Martine de Vos (VU) *Interpreting natural science spreadsheets*
36 Yuanhao Guo (UL) *Shape Analysis for Phenotype Characterisation from High-throughput Imaging*
37 Alejandro Montes Garcia (TUE) *WiBAF: A Within Browser Adaptation Framework that Enables Control over Privacy*
38 Alex Kayal (TUD) *Normative Social Applications*
39 Sara Ahmadi (RUN) *Exploiting properties of the human auditory system and compressive sensing methods to increase noise robustness in ASR*
40 Altaf Hussain Abro (VUA) *Steer your Mind: Computational Exploration of Human Control in Relation to Emotions, Desires and Social Support For applications in human-aware support systems*
41 Adnan Manzoor (VUA) *Minding a Healthy Lifestyle: An Exploration of Mental Processes and a Smart Environment to Provide Support for a Healthy Lifestyle*
42 Elena Sokolova (RUN) *Causal discovery from mixed and missing data with applications on ADHD datasets*
43 Maaike de Boer (RUN) *Semantic Mapping in Video Retrieval*
44 Garm Lucassen (UU) *Understanding User Stories - Computational Linguistics in Agile Requirements Engineering*
45 Bas Testerink (UU) *Decentralized Runtime Norm Enforcement*
46 Jan Schneider (OU) *Sensor-based Learning Support*
47 Jie Yang (TUD) *Crowd Knowledge Creation Acceleration*
48 Angel Suarez (OU) *Collaborative inquiry-based learning*

2018

1 Han van der Aa (VUA) *Comparing and Aligning Process Representations*
2 Felix Mannhardt (TUE) *Multi-perspective Process Mining*
3 Steven Bosems (UT) *Causal Models For Well-Being: Knowledge Modeling, Model-Driven Development of Context-Aware Applications, and Behavior Prediction*
4 Jordan Janeiro (TUD) *Flexible Coordination Support for Diagnosis Teams in Data-Centric Engineering Tasks*
5 Hugo Huurdeman (UVA) *Supporting the Complex Dynamics of the Information Seeking Process*
6 Dan Ionita (UT) *Model-Driven Information Security Risk Assessment of Socio-Technical Systems*
7 Jieting Luo (UU) *A formal account of opportunism in multi-agent systems*
8 Rick Smetsers (RUN) *Advances in Model Learning for Software Systems*
9 Xu Xie (TUD) *Data Assimilation in Discrete Event Simulations*
10 Julienka Mollee (VUA) *Moving forward: supporting physical activity behavior change through intelligent technology*
11 Mahdi Sargolzaei (UVA) *Enabling Framework for Service-oriented Collaborative Networks*
12 Xixi Lu (TUE) *Using behavioral context in process mining*
13 Seyed Amin Tabatabaei (VUA) *Computing a Sustainable Future*
14 Bart Joosten (UvT) *Detecting Social Signals with Spatiotemporal Gabor Filters*
15 Naser Davarzani (UM) *Biomarker discovery in heart failure*
16 Jaebok Kim (UT) *Automatic recognition of engagement and emotion in a group of children*
17 Jianpeng Zhang (TUE) *On Graph Sample Clustering*
18 Henriette Nakad (UL) *De Notaris en Private Rechtspraak*
19 Minh Duc Pham (VUA) *Emergent relational schemas for RDF*
20 Manxia Liu (RUN) *Time and Bayesian Networks*
21 Aad Slootmaker (OUN) *EMERGO: a generic platform for authoring and playing scenario-based serious games*
22 Eric Fernandes de Mello Araujo (VUA) *Contagious: Modeling the Spread of Behaviours, Perceptions and Emotions in Social Networks*
23 Kim Schouten (EUR) *Semantics-driven Aspect-Based Sentiment Analysis*
24 Jered Vroon (UT) *Responsive Social Positioning Behaviour for Semi-Autonomous Telepresence Robots*
25 Riste Gligorov (VUA) *Serious Games in Audio-Visual Collections*
26 Roelof Anne Jelle de Vries (UT) *Theory-Based and Tailor-Made: Motivational Messages for Behavior Change Technology*
27 Maikel Leemans (TUE) *Hierarchical Process Mining for Scalable Software Analysis*
28 Christian Willemse (UT) *Social Touch Technologies: How they feel and how they make you feel*
29 Yu Gu (UvT) *Emotion Recognition from Mandarin Speech*
30 Wouter Beek (VU) *The "K" in "semantic web" stands for "knowledge": scaling semantics to the web*

2019

1 Rob van Eijk (UL) *Web privacy measurement in real-time bidding systems. A graph-based approach to RTB system classification*
2 Emmanuelle Beauxis Aussalet (CWI, UU) *Statistics and Visualizations for Assessing Class Size Uncertainty*

3 Eduardo Gonzalez Lopez de Murillas (TUE) *Process Mining on Databases: Extracting Event Data from Real Life Data Sources*
4 Ridho Rahmadi (RUN) *Finding stable causal structures from clinical data*
5 Sebastiaan van Zelst (TUE) *Process Mining with Streaming Data*
6 Chris Dijkshoorn (VU) *Nichesourcing for Improving Access to Linked Cultural Heritage Datasets*
7 Soude Fazeli (TUD) *Recommender Systems in Social Learning Platforms*
8 Frits de Nijs (TUD) *Resource-constrained Multi-agent Markov Decision Processes*
9 Fahimeh Alizadeh Moghaddam (UVA) *Self-adaptation for energy efficiency in software systems*
10 Qing Chuan Ye (EUR) *Multi-objective Optimization Methods for Allocation and Prediction*
11 Yue Zhao (TUD) *Learning Analytics Technology to Understand Learner Behavioral Engagement in MOOCs*
12 Jacqueline Heinerman (VU) *Better Together*
13 Guanliang Chen (TUD) *MOOC Analytics: Learner Modeling and Content Generation*
14 Daniel Davis (TUD) *Large-Scale Learning Analytics: Modeling Learner Behavior & Improving Learning Outcomes in Massive Open Online Courses*
15 Erwin Walraven (TUD) *Planning under Uncertainty in Constrained and Partially Observable Environments*
16 Guangming Li (TUE) *Process Mining based on Object-Centric Behavioral Constraint (OCBC) Models*
17 Ali Hurriyetoglu (RUN) *Extracting actionable information from microtexts*
18 Gerard Wagenaar (UU) *Artefacts in Agile Team Communication*
19 Vincent Koeman (TUD) *Tools for Developing Cognitive Agents*
20 Chide Groenouwe (UU) *Fostering technically augmented human collective intelligence*
21 Cong Liu (TUE) *Software Data Analytics: Architectural Model Discovery and Design Pattern Detection*
22 Martin van den Berg (VU) *Improving IT Decisions with Enterprise Architecture*
23 Qin Liu (TUD) *Intelligent Control Systems: Learning, Interpreting, Verification*
24 Anca Dumitrache (VU) *Truth in Disagreement - Crowdsourcing Labeled Data for Natural Language Processing*
25 Emiel van Miltenburg (VU) *Pragmatic factors in (automatic) image description*
26 Prince Singh (UT) *An Integration Platform for Synchromodal Transport*
27 Alessandra Antonaci (OUN) *The Gamification Design Process applied to*Massive) Open Online Courses
28 Esther Kuindersma (UL) *Cleared for take-off: Game-based learning to prepare airline pilots for critical situations*
29 Daniel Formolo (VU) *Using virtual agents for simulation and training of social skills in safety-critical circumstances*
30 Vahid Yazdanpanah (UT) *Multiagent Industrial Symbiosis Systems*
31 Milan Jelisavcic (VU) *Alive and Kicking: Baby Steps in Robotics*
32 Chiara Sironi (UM) *Monte-Carlo Tree Search for Artificial General Intelligence in Games*
33 Anil Yaman (TUE) *Evolution of Biologically Inspired Learning in Artificial Neural Networks*
34 Negar Ahmadi (TUE) *EEG Microstate and Functional Brain Network Features for Classification of Epilepsy and PNES*
35 Lisa Facey-Shaw (OUN) *Gamification with digital badges in learning programming*
36 Kevin Ackermans (OUN) *Designing Video-Enhanced Rubrics to Master Complex Skills*
37 Jian Fang (TUD) *Database Acceleration on FPGAs*
38 Akos Kadar (OUN) *Learning visually grounded and multilingual representations*

2020

1 Armon Toubman (UL) *Calculated Moves: Generating Air Combat Behaviour*
2 Marcos de Paula Bueno (UL) *Unraveling Temporal Processes using Probabilistic Graphical Models*
3 Mostafa Deghani (UvA) *Learning with Imperfect Supervision for Language Understanding*
4 Maarten van Gompel (RUN) *Context as Linguistic Bridges*
5 Yulong Pei (TUE) *On local and global structure mining*
6 Preethu Rose Anish (UT) *Stimulation Architectural Thinking during Requirements Elicitation - An Approach and Tool Support*
7 Wim van der Vegt (OUN) *Towards a software architecture for reusable game components*
8 Ali Mirsoleimani (UL) *Structured Parallel Programming for Monte Carlo Tree Search*
9 Myriam Traub (UU) *Measuring Tool Bias and Improving Data Quality for Digital Humanities Research*
10 Alifah Syamsiyah (TUE) *In-database Preprocessing for Process Mining*
11 Sepideh Mesbah (TUD) *Semantic-Enhanced Training Data AugmentationMethods for Long-Tail Entity Recognition Models*

12 Ward van Breda (VU) *Predictive Modeling in E-Mental Health: Exploring Applicability in Personalised Depression Treatment*
13 Marco Virgolin (CWI) *Design and Application of Gene-pool Optimal Mixing Evolutionary Algorithms for Genetic Programming*
14 Mark Raasveldt (CWI/UL) *Integrating Analytics with Relational Databases*
15 Konstantinos Georgiadis (OUN) *Smart CAT: Machine Learning for Configurable Assessments in Serious Games*
16 Ilona Wilmont (RUN) *Cognitive Aspects of Conceptual Modelling*
17 Daniele Di Mitri (OUN) *The Multimodal Tutor: Adaptive Feedback from Multimodal Experiences*
18 Georgios Methenitis (TUD) *Agent Interactions & Mechanisms in Markets with Uncertainties: Electricity Markets in Renewable Energy Systems*
19 Guido van Capelleveen (UT) *Industrial Symbiosis Recommender Systems*
20 Albert Hankel (VU) *Embedding Green ICT Maturity in Organisations*
21 Karine da Silva Miras de Araujo (VU) *Where is the robot?: Life as it could be*
22 Maryam Masoud Khamis (RUN) *Understanding complex systems implementation through a modeling approach: the case of e-government in Zanzibar*