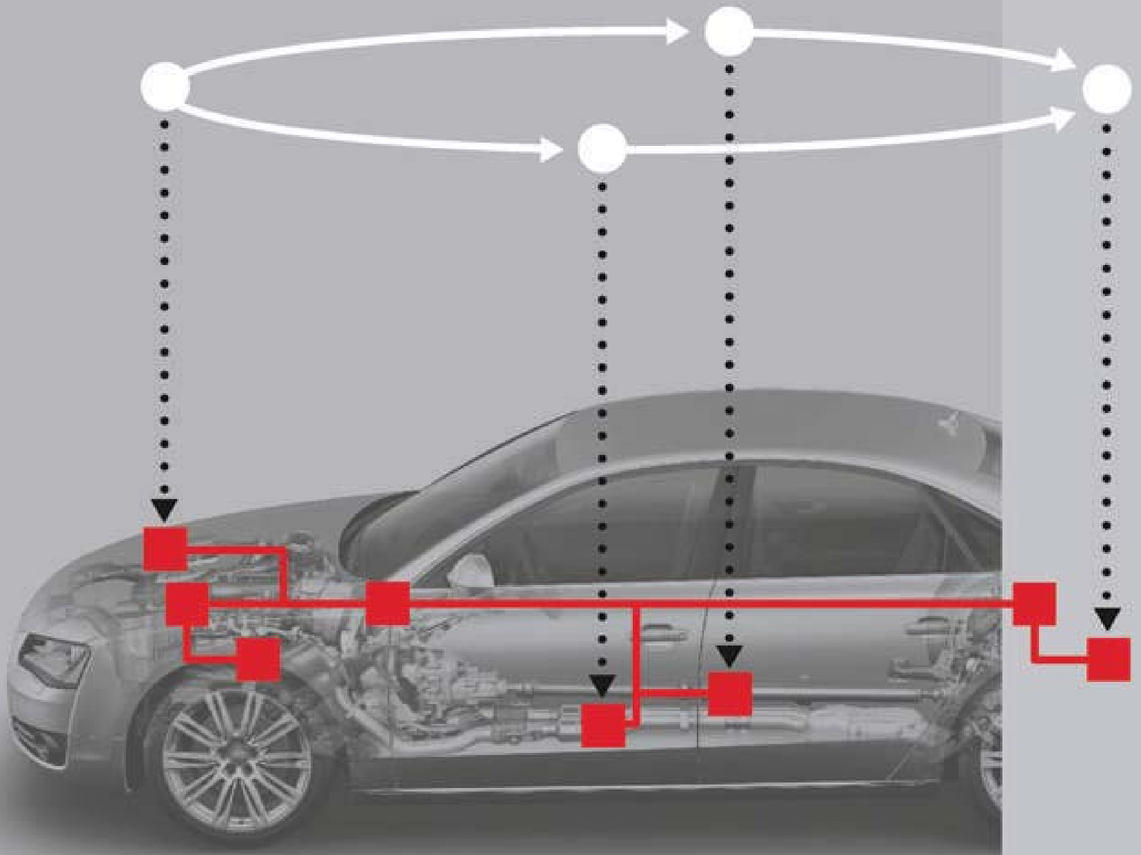
Audi
Dissertationsreihe

Modeling, Analysis, and Optimization of Automotive Networks

Martin Lukasiewycz



Cuvillier Verlag Göttingen

INI.FAU
INGOLSTADT INSTITUTE DER FRIEDRICH-ALEXANDER-UNIVERSITÄT ERLANGEN-NÜRNBERG

Modeling, Analysis, and Optimization of Automotive Networks

Der Technischen Fakultät der
Universität Erlangen-Nürnberg
zur Erlangung des Grades

D O K T O R - I N G E N I E U R

vorgelegt von

Martin Lukasiewycz

Erlangen 2010

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliographie; detaillierte bibliografische Daten sind im Internet über http://dnb.d-nb.de abrufbar.
1. Aufl. - Göttingen: Cuvillier, 2010
Zugl.: Erlangen-Nürnberg, Univ., 2010

978-3-86955-414-3

Audi-Dissertationsreihe, Band 36

Als Dissertation genehmigt von der Technischen Fakultät der Universität Erlangen-Nürnberg

Tag der Einreichung:	04.06.2010
Tag der Promotion:	28.07.2010
Dekan:	Prof. Dr.-Ing. Reinhard German
Berichterstatter:	Prof. Dr.-Ing. Jürgen Teich
	Prof. Dr. sc. Samarjit Chakraborty

978-3-86955-414-3

Acknowledgments

I would like to express my deepest gratitude to my supervisor Prof. Dr.-Ing. Jürgen Teich and my sponsor AUDI AG which made this thesis possible. It is also a pleasure to thank Dr.-Ing. Uwe Koser for his encouragement and his efforts regarding the cooperation between the University of Erlangen-Nuremberg and the AUDI AG in Ingolstadt.

The past three years gave me the opportunity to get in touch with many people in both academia and industry who were inspiring, supportive, or simply entertaining. In this short acknowledgment, most of them remain unnamed but they should be certain that they are not unnoticed. In particular, I thank my friend and colleague Michael Glaß for his kind support and his help with numerous publications that pave the long road towards this thesis. I also owe my gratitude to my friend and supervisor at AUDI Richard Regler for his patience, confidence, and encouragement while giving me valuable insights in the automotive domain.

Contents

1 Introduction

Since many years, the automotive industry is an important key driver in embedded system design and development. The vast majority of current technical innovations in modern vehicles such as advanced driver assistance systems or multimedia systems are realized or supported by electronic solutions. As a result, the amount of software and hardware in modern vehicles dramatically increased. Numerous distributed functions carry out safety, comfort, and driver-assistance functionality. In general, this functionality is developed separately and provided as hardware either by the vehicle manufacturer or a supplier. Here, the basic hardware unit is the Electronic Control Unit (ECU), i.e., a self-contained computer system often comprising its own operating system. Within an automotive network, the ECUs are interconnected via buses and gateways to enable the collaborative functionality.

As a result of the increased functionality, the number of ECUs has increased dramatically for top-of-the-range vehicles. The distribution of ECUs that are mounted either in protected mounting spaces or at their places of activity requires an integration of a complex bus system. Automotive networks are characterized by their heterogeneity comprising different bus systems such as Control Area Network (CAN) [CAN], Local Interconnect Network (LIN) [LIN], Media Oriented Systems Transport (MOST) [MOS] or FlexRay [FC05]. Each of these bus systems was developed for the automotive domain and answers its special purpose. The most common bus system for the communication between ECUs is the event-triggered CAN bus as well as the FlexRay bus that contains both a time-triggered and an event-triggered segment and allows a significantly higher data rate than the CAN bus. The MOST bus establishes a communication via fiber at a high data rate and is particularly used for multimedia applications. Operating at a low data rate, the LIN bus system is a time-triggered bus used for the connection of sensors and actuators to ECUs. Commonly, the bus systems are mutually interconnected via one or more gateways resulting in a heterogeneous topology.

The distributed functions implemented on the ECUs establish a communication by messages that are generally periodic or defined by a minimal inter-arrival time. In particular, critical functions such as airbag or engine functions impose severe requirements on the underlying system for a guaranteed functionality. Moreover, upcoming x-by-wire systems that aim to replace mechanical or hydraulic systems will increase the data traffic of future networks even further.

Due to the complexity and heterogeneity of state-of-the-art automotive ECU networks, the design of these networks becomes a challenging task. Currently, several essential design tasks such as the topology determination, function distribution, message routing and priority assignment of tasks and messages are performed separately and manually. This procedure has several drawbacks as it tends to be time-consuming, error-prone and might result in suboptimal implementations. The design of automotive systems has not only to consider the vast number of design decisions but also a high number of generally conflicting objectives such as monetary costs, energy consumption, robustness, or reliability. Thus, an automatic design approach becomes necessary to relieve the designer from the burden to optimize this network hand.

There exist some commercial tools to support the designer in the automotive domain, e.g., PREEvision [PRE], SymTA/S [HJRE04], or chronSIM [chr]. PREEvision is a widely used tool in the automotive domain to document and model the system. It lacks an automatic optimization of the discussed essential design tasks. The design tools SymTA/S and chronSIM allow to verify and test the real-time properties of functions either by a formal approach or simulation, respectively. Tools such as SymTA/S are only capable of optimizing the priorities of messages and tasks but do not optimize the system by any other respect. In contrast, the work at hand proposes an efficient optimization approach for automotive networks that concurrently performs a topology determination, function distribution, message routing and priority assignment of tasks and messages. The optimization considers multiple objectives such as monetary costs, energy consumption, end-to-end latencies, robustness, and reliability. Additionally, this work contributes a scheduling approach for the FlexRay bus that outperforms known approaches by the quality of the results in terms of number of allocated slots as well as by the significantly decreased runtime required to obtain a schedule.

The major contributions and achievements of this thesis are explained in the following paragraphs in more details.

1.1 Meta-heuristic Optimization of Constrained Combinatorial Problems

In the work at hand, a design space exploration that may automatically optimize automotive networks is proposed. This optimization requires a more flexible and efficient approach than is achievable by either Integer Linear Programming (ILP) or meta-heuristic approaches such as an Evolutionary Algorithm (EA). While an ILP is restricted to a single linear objective function, the meta-heuristic optimization performs not well in the presence of a high number of stringent constraints. To overcome these drawbacks, a novel constraint handling technique in meta-heuristic optimization is presented. This feasibility-preserving approach is a hybrid solution that combines the benefits of ILP and meta-heuristic optimization.

The experimental results of several random test cases from various domains give evidence of the benefits of this technique. Compared to a common penalty or repair approach, respectively, the proposed feasibility-preserving approaches show a better convergence for all test cases. In addition to that, the feasibility-preserving approaches even perform well on test cases where the common approaches fail to find even a single feasible solution.

This novel feasibility-preserving optimization approach and the detailed experimental results are presented in Chapter 2.

1.2 Design Space Exploration of Networked Embedded Systems

The proposed design space exploration may concurrently optimize the topology, function distribution, message routing, and parameters. This graph-based design space exploration is based on the Y-chart approach that maps an application, i.e., distributed functions, to an architecture such as ECUs, sensors, actuators, and buses. A model is introduced that is capable of modeling multi-hop and multicast communication as it is required in the automotive domain. In order to leverage the proposed optimization approach, a binary encoding of the design space exploration becomes necessary. The work at hand presents an efficient binary encoding of the model for networked embedded systems including a preprocessing algorithm to restrict the search space effectively. The experimental results give evidence of a fast convergence towards the optimal implementations also for large and complex problems with multiple objectives. As a result, the presented approach is superior compared to optimization methods known from literature.

Real-time and performance properties are essential for the correct functionality of distributed functions in automotive systems. This work presents a compositional timing model that takes advantage of the seamless combination of different analysis techniques. Based on this generalized formal timing analysis technique, a fine-grained fixed point iteration is proposed to deal efficiently with cyclic dependencies. Since the presented fine-grained approach always requires less computational steps than existing global approaches, it is capable of delivering performance metrics such as end-to-end delays very efficiently, and is therefore more scalable. In particular, it is proven that the fine-grained approach scales linearly compared to global approaches with a polynomial computational effort. An efficient approach is proposed that is able to break cyclic dependencies in the analysis, such that the fixed point iteration may not be applied globally, but can be applied separately to different subparts of the communication architecture.

The increasing complexity and number of electronic solutions in automotive systems that carry out critical functionality such as airbag or engine functions call for a formal reliability analysis and optimization technique. The work at hand presents a system-level design methodology for automotive networks that is capable of analyzing the reliability as well as exploiting existing data redundancy to increase reliability. The presented approach not only supports a reliability-aware design from scratch, but also enables the redesign of existing systems to increase the reliability at the expense of a minimal communication overhead. Several algorithms are proposed in order to automatically identify inherent data redundancy and an extension of the design space exploration model and encoding is presented that allows to exploit the revealed data redundancy. A symbolic analysis is proposed that quantifies the reliability of a system, enabling the usage of reliability as one of multiple conflicting optimization objectives.

Early decisions in embedded system design may be revised in later stages resulting in additional costs. This work presents a methodology to evaluate and optimize the robustness of an embedded system in terms of invariability in case of design revisions. A method that quantifies the expected additional costs as the robustness value is proposed. Since the determination of the robustness based on arbitrary revisions is computationally expensive, an efficient set-based approach that uses a symbolic encoding is presented. Moreover, a methodology for the integration of the optimization of the robustness in the design space exploration is proposed. Based on an external archive that accepts also near-optimal solutions, this robustness-aware optimization is efficient since it does not require additional function evaluations as previous approaches.

Several realistic case studies give evidence of the benefits of the proposed methodologies. These case studies comprise the ECU design of a Motion-JPEG decoder, automotive network integration, and an automotive network exploration.

The design space exploration of networked embedded systems including the extended exploration model, the improved real-time analysis technique, the reliability analysis, and the robust design methodology is presented in Chapter 3. The chapter also presents detailed experimental results based on synthetic and realistic test cases.

1.3 FlexRay Scheduling

The novel FlexRay [FC05] bus is going to sustainably change the design paradigm in the automotive domain. The currently prevailing CAN bus in the automotive domain is event-triggered and restricted to a low data rate. In contrast, the FlexRay bus offers both a static and dynamic segment with a high data rate. While the dynamic event-triggered segment is mostly used for maintenance and diagnosis data, the static time-triggered segment is used for critical application data. It is projected that first x-by-wire systems will be implemented on the FlexRay bus. However, the FlexRay protocol requires a large set of parameters but also a predefined scheduling of messages. In particular, scheduling the static segment in compliance with AUTomotive Open System ARchitecture (AUTOSAR) specification [AUT08] that provides a multiplexing scheme to extended to utilization of the bus becomes a challenging task.

The work at hand presents a FlexRay scheduling approach for the static segment based on two-dimensional bin packing. For this purpose, a transformation of the slot packing to bin packing and vice versa is presented. A proof shows the correctness of the transformation. For the bin packing a common greedy heuristic might be applied. In order to achieve an optimal schedule in terms of a minimal number of allocated slots, an ILP approach is presented that takes advantage of the problem specific constraints. This efficient encoding is further improved by additional constraints that minimize the search space effectively. In order to determine and improve the extensibility of given FlexRay schedules, an extensibility metric as well as a heuristic for the optimization of partially allocated slot is presented.

The experimental results show that the runtime of the transformation is negligible. For a given case study, the proposed approach is compared to the results obtained by a commercial tool. The proposed approach is four orders of magnitude faster and is capable of delivering the optimal schedule in terms of the allocated slots such that the result from the commercial tool is improved by two slots. The essentially improved runtime allows integrating this scheduling approach into an efficient design space exploration.

This FlexRay scheduling approach including the detailed experimental results is presented in Chapter 4.

2 Meta-heuristic Optimization of Constrained Combinatorial Problems

2.1 Introduction

This chapter introduces a novel optimization approach for discrete multi-objective optimization problems. This efficient optimization is essential for an effective Design Space Exploration (DSE) in the automotive domain. In particular, the optimization problems in the automotive domain are characterized by a huge search space due to the problem size and complexity as well as by stringent constraints due to several domain-specific requirements. In the following, the proposed efficient optimization is introduced and compared on a set of test cases from different domains before it is applied successfully to the DSE of automotive networks in Chapter 3.

Meta-heuristic algorithms are successfully applied to many complex optimization problems. In particular, some of these meta-heuristic optimization algorithms like Evolutionary Algorithms (EAs) perform very well on multi-objective problems. A major shortcoming of these meta-heuristic optimization algorithms is the missing of capability of innately handling arbitrary constraints. Though several generic and specific methods were researched to overcome this drawback, these methods tend to perform badly in case of a general constrained combinatorial problem where the search space is discrete and linearly constrained. Such a *constrained combinatorial problem* is defined as follows:

Definition 2.1 (Constrained Combinatorial Problem)

$$\begin{aligned}&\text{minimize } f(\mathbf{x})\\&\text{subject to:}\\&\quad A\mathbf{x} \leq b \textit{ with } \mathbf{x} \in \{0,1\}^n, A \in \mathbb{Z}^{m,n}, b \in \mathbb{Z}^m\end{aligned}$$

The objective function f might be multi-dimensional and non-linear. In single-objective optimization, the feasible set of solutions is totally ordered, whereas in multi-objective optimization problems, the feasible set is only partially ordered and, thus, there is generally not only one global optimum, but a set of *Pareto-optimal solutions*. A Pareto-optimal solution is better in at least one objective when compared to any other feasible solution. The search space $X = \{0,1\}^n$ is restricted to binary values, but allows integer values by a binary encoding. The *feasible search space* $X_f \subseteq X$ is restricted by a set of constraints which are subsumed in the stated matrix inequation $A\mathbf{x} \leq b$. Thus, the constraints have to be linear or linearizable.

In case of a relatively small feasible search space, common constraint-handling methods like penalty functions or local repair algorithms are more focused on the search for feasible solutions than on the optimization of the objectives. Figure 2.1 illustrates the shortcomings of a variation of a feasible solution in a constrained search space where the resulting solutions might become infeasible. As a result, only a slow convergence towards the optimal solutions is reached typically. In some cases it might even happen that the meta-heuristic optimization algorithm is not able to find even a single feasible solution. On the other hand, using exact approaches like Integer Linear Programming (ILP) is prohibited by the condition that the objective function is multi-dimensional and non-linear.

To overcome the drawbacks of known optimization methods for the constrained combinatorial problem, a novel approach is proposed in this chapter. This hybrid approach combines the benefits of ILP and meta-heuristic optimization methods, particularly EAs. Since the constraints in Definition 2.1 are restricted to binary variables, a backtracking-based ILP solver might be used to find feasible solutions. This so-called Pseudo-Boolean (PB) solver is incorporated into the meta-heuristic optimization process to enable a constraint handling and preserve the feasibility of the solutions.

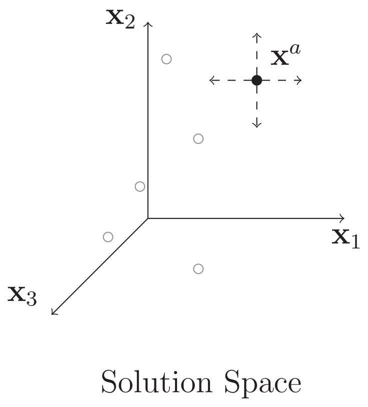


Figure 2.1: Illustration of a common operator that varies a solution $\mathbf{x}^a$ in a constrained search space. The circles correspond to feasible solutions. The variation of solution $\mathbf{x}^a$ may result in infeasible solutions.

Two basically different approaches are proposed that allow integrating a PB solver in the optimization process. First, a decoding approach is presented where the meta-heuristic is used to vary the branching strategy of a PB solver instead of varying the solutions directly [LGHT07]. As a result, the PB solver is used with the branching strategy to obtain feasible solutions. The second approach presents feasibility-preserving operators that are used by the optimization algorithm to vary the solutions inside the feasible search space [LGHT08b, LGT08]. For each operator an individual scheme for the branching strategy for PB solver is proposed. In case of an unconstrained problem, these operators degrade to the known bitwise operators. In particular, this work presents neighborhood, mutation, and crossover operators.

Several test cases give evidence of the benefits of the feasibility-preserving optimization approaches. A random set of single-objective test cases is selected from the PB Evaluation [MR09]. These test cases allow a fair comparison of the feasibility-preserving techniques with a penalty approach based on the convergence towards the optimal objective value. Additionally, two-dimensional optimization problems show the applicability in the multi-objective domain.

An introduction of the meta-heuristic optimization of the constrained combinatorial problem is given as follows:

- An extensive presentation of related work. (cf. Section 2.2)
- A detailed introduction on the PB problem. (cf. Section 2.3)

In summary, this chapter provides the following contributions to the constrained combinatorial problem as defined in Definition 2.1:

- A decoding scheme based on a PB solver that ensures the feasibility of solutions. (cf. Section 2.4)
- An operator scheme based on a PB solver that preserves the feasibility of solutions. (cf. Section 2.5)
- A meaningful set of test cases that gives evidence of the superiority of the proposed methods compared to known constraint handling methods in meta-heuristic optimization. (cf. Section 2.6)

2.2 Related Work

This section discusses existing work related to meta-heuristic optimization and constraint handling methods known from literature. Finally, known hybrid optimization algorithms that combine exact and heuristic algorithms are discussed.

2.2.1 Meta-heuristic Optimization

Meta-heuristic optimization methods are commonly used for complex problems where common optimization techniques like Linear Programming (LP), Quadratic Programming (QP), Geometric Programming (GP), etc. are not applicable due to their restrictive expressiveness. This is often the case if the problem has multiple non-linear objectives or constraints as well as a complex underlying problem representation.

The domain of meta-heuristic optimization comprises methods such as Evolutionary Algorithm (EA) approaches [Bäc96], Simulated Annealing (SA) [KGV83, Čer85], and Particle Swarm Optimization (PSO) [KE95]. The abovementioned meta-heuristic optimization algorithms are inspired by nature and, therefore, also embraced by the term Evolutionary Computation (EC) [Fog95].

Evolutionary Algorithms

An EA is a population-based optimization algorithm inspired by biological evolution. The first EAs were developed independently by different research groups around the world in the 1960s. Lawrence J. Fogel coined the term Evolutionary Programming in 1962 [Fog62, FOW66] for his evolutionary approach to solve prediction models. Also in 1962, John H. Holland initiated the Genetic Algorithm (GA) approaches [Hol62] resulting in a pioneering book published in 1975 [Hol75]. His work was motivated by the development of robust adaptive systems. The work on the Evolution Strategy (ES) started in the 1960s and was further developed in the 1970s by Ingo Rechenberg and Hans-Paul Schwefel [Rec71, BS02]. The ES approaches were used to solve continuous parameter optimization problems. All these approaches share the common idea of using *reproduction* and *natural selection*. Therefore, these approaches are commonly embraced by the term EA. Numerous books have been published on the EA topic such as [Dav91, Mic96, BNKF98].

The main procedure of an EA is based on the reproduction and selection that are performed alternately in an iterative process to optimize a given objective. The reproduction creates new individuals from the current population using the *mutation* and *crossover* operators. These crossover and mutation operators are problem specific. There exist numerous general-purpose crossover and mutation operators for real, integer, binary values and also for problem specific data-structures. The task of the selection is to remove the worst individuals to ensure a convergence of the algorithm towards the optimal solutions. For single-objective optimization this might be a pure elitism selection that always removes the worst individuals or a probabilistic method like the roulette wheel selection.

In recent years, huge efforts were made to adapt the selection to multi-objective problems. The best known and commonly used algorithms for multi-objective selection are the Strength Pareto Evolutionary Algorithm 2 (SPEA2) [ZT99, ZLT02], the Non-dominated Sorting Genetic Algorithm II (NSGA-II) [SD94, DAPM00], and the Indicator Based Evolutionary Algorithm (IBEA) [ZK04].

Simulated Annealing

SA is an optimization algorithm inspired by the annealing process in metallurgy. It is a further development of the Metropolis algorithm [MRR$^+$53] and was developed independently by Kirkpatrick et al. in 1983 [KGV83] and Černỳ in 1985 [Čer85].

The algorithm varies a single solution by a neighborhood operator. The common neighborhood operator for binary problems is a bit flip or the sampling from a normal distribution for real-valued problems. However, also problem-specific neighbor operators were studied and successfully applied to arbitrary problems. The common SA improves a single solution iteratively. A new solution is either accepted or rejected based on a probability value that is calculated using a continuously decreasing temperature function.

The general SA is a single-objective optimization algorithm. However, also multi-objective variants were studied [TK07, BSMD08]. A further extension of the SA is the Tabu Search (TS) [Glo89, Glo90] where each found solution is excluded from the search space to enable a faster convergence towards the optimal solution.

Particle Swarm Optimization

PSO is an optimization algorithm based on swarm intelligence using social-psychological principles. It was first published in 1995 by James Kennedy and Russell C. Eberhart [KE95].

The main procedure is based on particles that are moving in the search space. As a result, the algorithm mainly targets continuous problems. A swarm consists of multiple particles that change their position on each iteration. One single particle is attracted by its local best position and the global best particle. Here, the quality of an particle depends on the objective function. There exist several variations including extensions for multi-objective optimization [CCPL04].

Miscellaneous methods

In addition to the abovementioned meta-heuristic optimization algorithms there exist other approaches which are outlined in the following. Differential Evolution (DE) [SP95] is an optimization approach tailored for continuous search spaces and shares several similarities with PSOs. Ant Colony Optimization (ACO) [DMC96] is restricted to optimize paths through graphs inspired by ant behavior. Meta-heuristic algorithms like Artificial Immune Systems (AIS) [FPAC94], Harmony Search (HS) [GKL01], and many others are tailored for specific optimization problems. Moreover, there exist various hybrid combinations of the discussed algorithms.

2.2.2 Constraint Handling

Though the meta-heuristic optimization algorithms generally aim to optimize unconstrained problems, several different approaches that aim to extend the optimization algorithms for constraint handling exist. These approach are discussed in detail in [MS96, CC02] and outlined in the following.

The most common approach to incorporate constraints into meta-heuristic optimization is the usage of penalty functions [SC95]. A constrained problem is transformed into an unconstrained problem such that a penalty function deteriorates the objective function depending on the violation of the constraints.

The basic implementation is based on static penalty functions. Here, the penalization of infeasible solutions is performed by adding a weighted distance to the feasible region to the objective value [RPLH89]. This requires an appropriate constant weight for each constraint that has to be determined manually. To overcome this drawback, dynamic penalty functions change these weights depending on the current iteration of the optimization algorithm [JH94, Ols94]. In particular, the weights are monotonically increasing such that the solutions are penalized more and more with a proceeding optimization. This ensures an exploration of the complete search space in the early phase of the optimization while eventually the solutions move to the feasible region. All penalty approaches share the disadvantage that without a proper parameter set they might get stuck in local optima or deliver too few feasible solutions. A more sophisticated method is the adaptive penalty function [BHAB97]. In this case, the penalty function is controlled by a feedback from the current search process. Nevertheless, also adaptive approaches require an appropriate parameter set.

Rejecting infeasible solutions requires no parameters and is computationally efficient since the determination of the objective functions is not required [Sch81]. However, if the feasible region is relatively small, this approach might take a very long time until a single feasible solution is found. A more sophisticated method that overcomes the drawbacks of rejecting infeasible solutions, is to put feasible over infeasible solutions [PS93]. In this case, the infeasible solutions are used to guide the algorithm towards the feasible search space by reducing the constraint violation. This approach is still computationally efficient similar to the rejection of infeasible solutions with the exception that the violation of all constraints has to be determined. Nevertheless, this approach might get stuck in local optima once a feasible solution is found.

Handling the constraints as additional objectives was studied and summarized in [HABRCC$^+$04]. Either the violation of each constraint is an additional objective [FF95] or the sum of the violation is a single additional objective [CT97]. The drawbacks of these approaches are the increased complexity of the problem by additional objectives and a slow convergence if the feasible region is relatively small compared to the entire search space.

For some problems there exists a special representation or operators that preserve the feasibility of the solutions. A popular example of this is the *Traveling Salesman Problem* [LKM$^+$99, GK00] where a permutation encodes the round trip and the operators preserve the permutations. Some combinatorial problems allow simple repair algorithms that restore the feasibility of the solutions. The *0/1 Knapsack Problem* [ZT99] is a common example where removing items from the knapsack repairs the solutions. For linear equality constraints, feasibility can also be preserved by eliminating decision variables [MN95]. If not all constraints are linear, this approach can at least be used to minimize the search space and exclude some infeasible regions. In fact, if a simple repair algorithm exists, also penalty functions guide the search fast towards the feasible region [KBH94]. In case the objectives are not calculated for infeasible solutions, this simple approach is nearly as effective as many problem specific greedy repair algorithms.

In [Bel97], a repair approach of infeasible solutions based on the *simplex* algorithm is presented. This approach is applied to real-valued problems and is only focused on satisfying the constraints such that the information from the repaired infeasible solution is discarded. This high degree of randomness results in a rather slow optimization convergence.

A further constraint handling approach for EAs is a custom decoding strategy. The information from the chromosome is decoded to the feasible search space. In [KM98], a decoding for continuous constrained optimization problems is done between a multi-dimensional cube and the feasible search space. However, this decoder is using a problem-specific mapping and, moreover, cannot be applied to discrete constrained problems.

In[CC02], Coello Coello outlines the prevailing difficulty of greedy repair algorithms and feasibility-preserving methods: