ON-LINE PARAMETER OPTIMIZATION STRATEGIES FOR METAHEURISTICS

Michael Affenzeller, Lukas Pöllabauer, Gabriel Kronberger, Erik Pitzer, Stefan Wagner, Stephan M. Winkler, A. Beham, M. Kofler

Upper Austria University of Applied Sciences
School for Informatics, Communications, and Media
Heuristic and Evolutionary Algorithms Laboratory
Softwarepark 11, 4232 Hagenberg, Austria

michael.affenzeller@fh-hagenberg.at , lukas.poellabauer@fh-hagenberg.at, gabriel.kronberger@heuristiclab.com, erik.pitzer@heuristiclab.com, stefan.wagner@heuristiclab.com, stephan.winkler@fh-hagenberg.at, andreas.beham@heuristiclab.com, monika.kofler@heuristiclab.com

ABSTRACT
In this paper we describe different aspects of parameter tuning strategies for meta-heuristic algorithms. In contrast to many automated parameter adjustment methods in the field, special attention is given to parameter tuning strategies which are able to be applied during the run of a meta-heuristics. The basic idea we are using for this approach steams from self adaptive evolution strategies and this paper discusses different adaptations to this idea in order to find out, to which extent this concept can be transformed to other meta-heuristics.

Keywords: Meta-heuristic Optimization, Evolutionary Algorithms, Parameter Tuning.

1. INTRODUCTION

Heuristic methods provide a reasonable tradeoff between achieved solution quality and required computing time, as they employ intelligent rules to scan only a fraction of a highly complex search space. Typical applications of heuristic methods can be found in production optimization; for example, heuristic algorithms are applied in machine scheduling and logistics optimization. For efficiently scanning such highly complex and exponentially growing search spaces, only heuristic methods can be considered for solving problems in dimensions which are relevant for real-world applications.

The step from heuristics to meta-heuristics is an essential one: While heuristics are often designed and tuned for some specific problem, meta-heuristics offer generic strategies for solving arbitrary problems. The implementation of concrete solution manipulation operators still depends on the problem representation, but the optimization strategy itself is problem-independent.

The success of meta-heuristics is based on an interplay between phases of diversification and intensification, but in order to achieve a beneficial equilibrium, fine-tuning is necessary for each problem instance depending on its fitness landscape characteristics (Affenzeller et al., 2009).

One of the most prominent representatives of meta-heuristics is the class of evolutionary algorithms (Eiben and Smith, 2003): New solution candidates (individuals) are generated by combining attributes of existing solution candidates (crossover) and afterwards they are slightly modified with a certain probability (mutation); parent individuals are chosen by means of nature inspired selection techniques (Holland, 1975). A second well-known example of a rather simple meta-heuristic is simulated annealing (Kirkpatrick et al., 1983): This approach is closely related to local search strategies such as hill climbing/descending and generates new solutions iteratively, starting from a usually randomly initialized solution. In contrast to simple hill climbing/descending, moves to worse solutions are permitted with a certain probability which decreases during the heuristic search process; by this means the algorithm first performs exploration (diversification), and later tends to focus on promising regions (intensification).

A multitude of other meta-heuristics has been described in the literature, such as for example particle swarm optimization (Eberhardt et al., 2001), tabu search (Glover, 1997) and iterated local search (Lourenco et al., 2003). The evolution of so many diverse meta-heuristics results from the fact that no single method outperforms all others for all possible problems. To be a bit more precise, the so-called No-Free-Lunch theorem postulates that a general-purpose universal optimization strategy is impossible and that the only way how one strategy can outperform another is to be more specialized to the structure of the tackled problem. The No-Free-Lunch theorem basically says that, given two arbitrary meta-heuristics (including random search), there always exist search spaces for which the first meta-heuristic will perform better than the second and vice versa.

This means that even for the most sophisticated meta-heuristic a fitness landscape can be constructed for

which it performs worse than ordinary random search. Therefore, it always takes qualified algorithm experts to select, parameterize and tune a meta-heuristic algorithm for a concrete application.

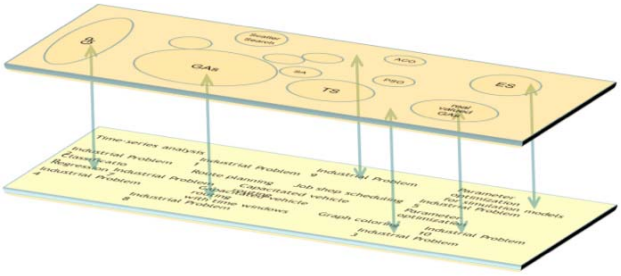

Figure 1: Schematic representation of the ordered relation between problem instances (lower layer) and potentially well suited meta-heuristics (upper layer).

This situation is illustrated in Figure 1 where the lower layer represents the problem instances with their associated fitness landscape characteristics and the upper layer shows the meta-heuristic methods under certain parameterizations. There exist some rough rules of thumb derived from empirical testing that indicate which meta-heuristics should be chosen for certain problem characteristics. However, picking an appropriate method for a certain problem instance is a non-trivial task. On the one hand, fitness landscape characteristics may change remarkably for different problem dimensions; on the other hand, the characteristics of a certain meta-heuristic may considerably vary for different parameter settings.

The problem of choosing an appropriate method and to find a beneficial set of parameters for a given optimization task is usually a challenging and time consuming task which takes an experienced algorithm expert. Alternatively, some automated approaches have been proposed that try chose the method or its parameter setting automatically. However, these approaches usually all suffer from enormous runtime consumption as it is necessary to run a set of parameter settings for the meta-heuristics to be optimized in order to evaluate only one single solution candidate of the optimization method which is responsible for optimizing the parameter tuning. The approach of optimizing the parameters of a meta-heuristics for a certain problem is often referred to as meta-optimization of meta-heuristics and is usually done with a meta-heuristics itself. Unfortunately, it is also no good idea to optimize the parameters applied to a small problem instance of the problem class to be solved in terms of parameter adjustment, as fitness landscape characteristics are usually totally different for different problem instance of the same problem class and therefore require different parameter settings. Summarizing all these considerations, it can be concluded that the application of meta optimization strategies for the parameter optimization task of a modern meta-heuristics applied to a challenging problem to be solved is a very time consuming task and therefore usually not applied in practice where up-to date hardware environment is already on its limits just for solving the problem to be solved.

In this paper we aim to introduce a general purpose lazy meta-optimization approach which is able to adapt certain parameter setting of a meta-heuristic algorithm during its run. The basic idea is inspired by the general idea, how self adaptive evolution strategies handle componentwise adjustment of the strategy parameters during the run (Schwefel, 1975). The main advantage of this strategy is, that, even if it is not so powerful than pure meta-optimization approaches, that it should be quite easily applicable to even quite time consuming problem instances as the runtime requirements are not much higher than in the case of a standard meta-heuristics.

The paper is organized as follows: Sections 2 and 3 describe the theoretical foundations for the new lazy meta-optimization approach which is introduced in section 4. In section 5 first results of the lazy meta-optimization are shown exemplarily for genetic algorithms with and without offspring selection (Affenzeller and Wagner, 2005). Finally, section 6 concludes the main characteristics of the proposed approach and touches on some ideas about future application areas of the proposed ideas.

2. META OPTIMIZATION

As stated in the introduction, the choice of a certain meta-heuristics as well as the parameter tuning process is highly dependent on the problem instance which is to be optimized. With the No-Free-Lunch theorem in mind there is also strong theoretical evidence that a universal solver meta-heuristics cannot be realized.

Meta meta-heuristics, in which meta-heuristics are used for the parameter tuning process of a meta-heuristics, has have already been discussed for a long period of time. In (Grefenstette, 1986), for example, a genetic algorithm has been used for tuning the parameters of a genetic algorithm.

A severe drawback of such approaches is that for the fitness evaluation of a high-level (meta-meta level) solution candidate a complete run of the low level algorithm has to be performed; considering the fact that reams of thousands of evaluations are usually required for achieving satisfying results and that each evaluation on the high level requires a full run on the low-level meta-heuristics it becomes easy to follow that this approach is still rather restricted to solve rather simple (low dimensional) problems. A crucial fact concerning this approach is also that it is not advantageous to identify good algorithm parameters on the basis of low dimensional problems and apply them on higher dimensional problems because the fitness landscape characteristics are drastically changing even for different problem instances of the same problem class.

A comprehensive review of parameter control strategies for evolutionary algorithms is stated in (Eiben et al., 1999).

A similar flavor which is driven by the availability of parallel hardware systems is given by hyper-heuristics (Burke et al., 2003). The goal here is, that hyper-heuristics should be able to adapt to the fitness landscape characteristics associated with a certain problem. However, the approach is quite different: Hyper-heuristics operate on a higher level of abstraction: they choose a certain meta-heuristics out from a set of available meta-heuristics depending on the actual performance which may be measured on the basis of CPU-time and the change in fitness. Even this approach seems to be quite simple and also quite efficient to handle especially in a parallel environment, it is not able to tune the parameters of the certain meta-heuristics themselves.

In the following sections we will propose a new attempt which may be seen as some kind of compromise of meta-meta heuristics and hyper heuristics. In this approach, the tuning of parameters should be done during the run of a single algorithm in way which is inspired by the way evolution strategies adaptively handle step-width regulation according to (Beyer and Schwefel, 2002).

3. SELF-ADAPTIVE ES

With the 1/5 success-rule, Rechenberg introduced a regulator for the mutation strength depending on the ratio of successful mutants in a certain generation (Rechenberg, 1973). If the ratio of successful mutants is high (>1/5) it is rather easy to achieve improvements and the optimum is considered to be far away. Therefore, the mutation strength (step-width) is increased. In case of too little successful mutants (<1/5) mutation strength is decreased because it is argued that the algorithm may jump around an optimum without detecting it.

Following basically the same idea, Schwefel introduced a self-adaptive step-width regulation which is applied individually to each dimension of the parameter vector for each solution candidate of the ES population (Schwefel, 1975). Technically, this is done in the following way:

The dimension of an *n*-dimensional parameter vector is doubled resulting in a parameter vector of dimension *2n*. The additional *n* dimensions store the actual standard deviation of the usually normal distributed mutation step-width for each dimension of the parameter vector.

$$X = ((x_1, x_2, \ldots x_n), (\sigma_1, \sigma_2, \ldots, \sigma_n))$$
$$x_i' = x_i + N(0, \sigma_i') \text{ (mutation)}$$

Consequently, also the mutation step-widths have to be mutated as well. This is usually done by adding some Gaussian noise around 0 with a constant term and a term which depends on the dimension.

The aspect of this adaptive parameter tuning strategy which is quite unique is the fact that the mutation step-widths σ_i are implicitly optimized even if we do not have an explicit fitness function for them. In other word this means that this strategy operates under the assumption that above average solution candidates are more likely to have emerged with advantageous parameters (step-width variances).

The main idea of the present contribution is to consider, how and to which extent this idea can be transferred to other meta-heuristics. The next two chapters exemplarily point out some concrete strategies about how such an ES-inspired lazy meta-optimization strategy may be transferred to genetic algorithms and give some preliminary results.

4. LAZY META-OPTIMIZATION

In this section we consider which parameter of a standard genetic algorithm (SGA), an island model parallel genetic algorithm (PGA), and an offspring selection genetic algorithm (OSGA) can be optimized implicitly during the run of the algorithm.

Basically, offspring selection is defined in the following way: After generating new solutions by crossover and mutation, these new solutions are inserted into the next generation's population only if they are better than their parents. The decision whether a child is considered better than its parents depends on the so-called comparison factor *cf* (*cf* ∈ [0, 1]) parameter: If *cf=0* then a child is already considered better if it surpasses the fitness of the worse parent, if *cf=1* then a child has to be better than both parents for being considered successful. Additionally, the parameter *success ratio* defines the ratio of successful individuals in the next population; the rest of the population is filled up with children that do not necessarily have to be successful. Details about this procedure can for example be found in (Affenzeller and Wagner, 2005) or in (Affenzeller et al., 2009).

For the considered GA variants the following parameter values have to be adjusted:

- Population size
- Selection operator
- Elitism
- Crossover operator
- Mutation operator
- Mutation rate
- Additionally for PGAs
  - Communication topology
  - Migration scheme
  - Migration rate
  - Migration interval
- Additionally for OSGAs
  - Comparision factor
  - Success ratio
  - Maximum selection pressure

The only parameters that can be considered in a sense that they were relevant for the evolvement of a certain individual are the choice of the crossover operator and the mutation operator.

Parameters like the mutation rate have to be analyzed with respect to the evolvement of the population.

So far we have analyzed combinations of different crossover and mutation operators which are stored together with the individual (the individual stores the operator by which it has been evolved). Due to the sexual recombination aspect, a local family tournament is applied when the two parent individuals are carrying different crossover operators in their parameter knapsack. If mutation is to be applied in the actual reproduction step, the family tournament has to be extended for up to four potential offspring solution candidates that may arise, if both parents have used different crossover and mutation operators.

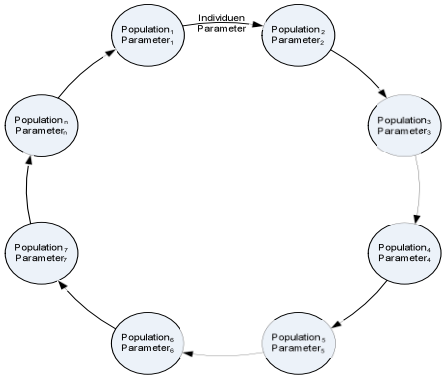

Figure 2: Migration of parameters (and) individuals in the topology of a unidirectional ring.

Concerning the online-optimization of strategy parameters that have to be interpreted on the population level several more challenging questions arise – especially in the context of genetic algorithms. In contrast to standard meta-optimization approaches, here we want to evaluate the actual potential of the population in terms of achievable solution quality during the run of the algorithm. Therefore, it becomes necessary to define some measure for the potential of the population with respect to achievable solution quality. Having in mind the building block theory for genetic algorithms (e.g. in Affenzeller et al, 2009) the achievable solution quality of a genetic algorithm mainly depends on the distribution of essential genetic information over the individuals in a certain population and how the available genetic operators are able to combine this essential genetic information to better and better solution candidates.

The evaluation function for the actual potential of a population may be defined in terms of a measure which calculates the relative ratio of successful offspring of a random sample taken from the actual population where an offspring is considered successful if its fitness is better than the fitness of the better two parents which can be evolved using the actual set of parameters and operators. However, the question remains how the parameters which are relevant for the evolvement of a population can be compared amongst each other. For this purpose we propose an architecture which is similar to a course grained parallel GA. The only difference to an island model GA is that the concept of migration is adapted for the exchange of advantageous parameter settings between the subpopulations as indicated in Figure 2.

5. RESULTS

In this section first results of lazy meta-optimization are shown for genetic algorithms considering the performance of different crossover operators for the *ch130* travelling salesman benchmark problem taken from the TSPLib. The experiments have been performed with the framework HeuristicLab 3.3[1] using the set of parameters shown in Table 1 and Table 2. A detailed description of the listed GA parameters and operators can be found in (Affenzeller et al., 2009).

Parameter	Value
Generations	2000
Population size	100
Selection operator	Proportional
Mutation operator	Inversion
Mutation rate	5%
Elitism strategy	1-elitism
Crossover operators	Family tournament using the following set of crossover operators: AbsolutePositionTopologicalCrossover OrderBasedCrossover PositionBasedCrossover CyclicCrossover OrderCrossover EdgeRecombinationCrossover MaximalPreservativeCrossover PariallyMatchedCrossover CosaCrossover

Table1: Parameters for the experiments shown in Figure 3 and Figure 4.

Figure 3 shows that when using a standard GA basically only two (ERX and Cosa) out of the nine available crossover operators turn out to be successful where ERX becomes even more successful in the final (almost converged) stage when the goal is to achieve still minor improvements.

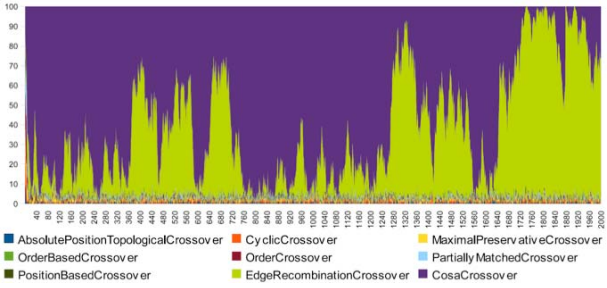

Figure 3: Distribution of crossover operators when using a standard GA.

[1] https://dev.heuristiclab.com/trac/hl/core

Parameter	Value
Generations	500
Success Ratio	1
Comparison Factor	1
Max. selection pressure	100

Table 2: Additional (offspring selection) parameters for the experiments shown in Fig. 4.

The remaining seven crossover operators turn out to be ineffective for the given problem and are therefore practically not used even if some survival strategy is implemented in order to avoid the total disappearance of an operator.

When using an offspring selection GA (additional parameters are given in Table 2) the dominance of single operators becomes even more evident and it is especially the ERX operator which extensively outperforms the other operators; only the Cosa operator can contribute in some stages of the algorithm.

Even if we can only show some snapshot results in this paper, the basic characteristics (dominance of ERX and Cosa) explained here are characteristic for other TSP experiments. When experimenting with other problem instances or even other problems the characteristics are different of course which is one of the main motivations for meta-meta or hyper-heuristics.

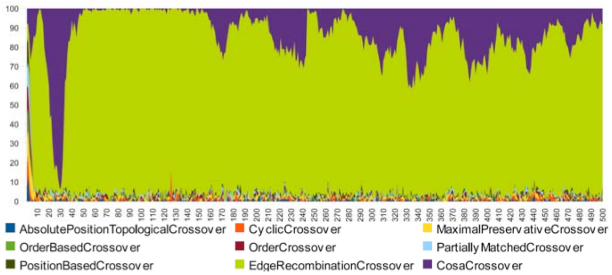

Figure 4: Distribution of crossover operators when using an offspring selection GA.

Concerning the experiments it has to be pointed out that the results shown here are only exemplarily and many other experimental setups are to be addressed in the future. Some suggestions are given in the conclusion. Furthermore, the main goal of the experiments of this paper was not focused on solution quality but rather on indicating the potential of the proposed lazy meta-optimization approach. However, the achieved results are close (within a 5% range) to the global optimal solution and comparable or better than with any single operator.

6. CONCLUSION

In this paper we have proposed several aspects for a lazy meta-optimization approach which is inspired by the implicit strategy parameter optimization of evolution strategies with adaptive step-width regulation according to (Schwefel, 1975). Concretely, it has been considered, which parameters of several genetic algorithm variants may be optimized following this strategy. First exemplary results of this approach have been shown.

However, many aspects should be analyzed in more detail: Open topics for future research in this field concern the mentioned family tournament. Alternatively also random operators of the two parent individuals could be used analyzing the effects concerning performance and expressiveness. Also the analysis of different migration strategies (for the exchange of parameters relevant for the performance of subpopulations) need to be studied in more detail.

Furthermore, the evaluation of the actual potential of a GA population is a challenging research topic of its own. As an alternative to the proposed potential evaluation, also a combined measure could be used considering the best and average individual fitness combined with a diversity potential.

ACKNOWLEDGMENTS

The work described in this paper was done within the Josef Ressel Centre for Heuristic Optimization *Heureka!* (http://heureka.heuristiclab.com/) sponsored by the Austrian Research Promotion Agency (FFG).

REFERENCES

Affenzeller, M. and Wagner, S., 2005. Offspring selection: A new self-adaptive selection scheme for genetic algorithms. *Adaptive and Natural Computing Algorithms*, Springer Computer Science, pp. 218-221.

Affenzeller, M., Winkler, S., Wagner, S., A. Beham, 2009. *Genetic Algorithms and Genetic Programming - Modern Concepts and Practical Applications*. Chapman & Hall/CRC. ISBN 978-1584886297.

Beyer, H-G. and Schwefel. *Evolution strategies – A Comprehensive Introduction*. Natural Computing 1(1). 2002. pp. 3 – 52.

Burke, E., Kendall, G., Newall, J., Hart, E., Ross P., Schulenburg S., 2003. Hyper-heuristics: An Emerging Direction in Modern Search Technology. In: F. Golver and G.A. Kochenberger, ed. *Handbook of Metaheuristics*. Kluwer Academic Publishers: pp. 457–474.

Eberhardt, R. C., Shi, Y, Kennedy, J., 2001. Swarm Intelligence. *The Morgan Kaufmann Series in Artificial Intelligence.*

Eiben, A.E, Hinterding, R., Michaelwicz Z. 1999. Parameter Control in Evolutionary Algorithms. *IEEE Transactions on Evolutionary Computation 3(2)*, IEEE Press, pp 124 - 141.

Eiben, A.E. and Smith, J.E. 2003. Introduction to Evolutionary Computation. *Natural Computing Series*, Springer-Verlag Berlin Heidelberg.

Glover, F. ,1997. Tabu Search and Adaptive Memory Programming – Advances, Applications, and Challenges. *Advances in Metaheuristics, Optimization and Stochastic Modeling Technologies* 7, Springer, pp. 1-75.

Grefenstette, J. ,1986. Optimization of control parameters for genetic algorithms. *IEEE Transactions on Systems, Man and Cybernetics 16 (1)*, IEEE Press, pp. 122-128.

Holland, J.H., 1975. *Adaption in Natural and Artifical Systems.* University of Michigan Press

Kirkpatrick, S., Gelatt Jr., C. D., Vecchi, M. P., 1983. Optimization by Simulated Annealing, *Science* 220(4598): 671-680.

Lourenco, H. R., Martin, O. C., Stützle, T., 2003 Iterated Local Search, Handbook of Metaheuristics (57), Kluwer 321 – 352.

Rechenberg, I., 1973. *Evolutionsstrategie.* Friedrich Frommann Verlag

Schwefel, H.-P., 1975. *Evolutionsstrategie und Numerische Optimierung.* Technische Universität Berlin, Fachbereich Verfahrenstechnik: Dr.-Ing., Dissertation.

Schwefel, H.-P., 1994. *Numerische Optimierung von Computer-Modellen mittels der Evolutionsstrategie.* Basel: Birkhäuser Verlag

AUTHORS BIOGRAPHIES

MICHAEL AFFENZELLER has published several papers, journal articles and books dealing with theoretical and practical aspects of evolutionary computation, genetic algorithms, and meta-heuristics in general. In 2001 he received his PhD in engineering sciences and in 2004 he received his habilitation in applied systems engineering, both from the Johannes Kepler University of Linz, Austria. Michael Affenzeller is professor at the Upper Austria University of Applied Sciences, Campus Hagenberg, and head of the Josef Ressel Center *Heureka!* at Hagenberg.

LUKAS PÖLLABAUER studied Software Engineering at the Upper Austrian University of Applied Sciences (Campus Hagenberg) and received his MSc in 2009. His research interests include heuristic optimization methods and parameter optimization of metaheuristics. He is currently employed as a software developer at DHL Express Austria.

GABRIEL KRONBERGER authored and co-authored numerous papers in the area of evolutionary algorithms, genetic programming, machine learning and data mining. Currently he is a research associate at the Research Center Hagenberg of the Upper Austria University of Applied Sciences working on data-based modeling algorithms for complex systems within the Josef-Ressel Centre for Heuristic Optimization Heureka!.

ERIK PITZER received his diploma in software engineering in 2004 at the Upper Austria University of Applied Sciences. Since 2004 he has been working as a research associate at the Research Center Hagenberg of the Upper Austria University of Applied Sciences and at the Decision Systems Group of Harvard Medical School in Boston. Erik Pitzer is currently working on his PhD thesis in the field of fitness landscape analysis for the design and analysis of meta-heuristic algorithms.

STEFAN WAGNER received his MSc in computer science in 2004 and his PhD in engineering sciences in 2009, both from Johannes Kepler University (JKU) Linz, Austria; he is professor at the Upper Austrian University of Applied Sciences (Campus Hagenberg). Dr. Wagner's research interests include evolutionary computation and heuristic optimization, theory and application of genetic algorithms, machine learning and software development.

STEPHAN M. WINKLER received his MSc in computer science in 2004 and his PhD in engineering sciences in 2008, both from Johannes Kepler University (JKU) Linz, Austria. His research interests include genetic programming, nonlinear model identification and machine learning. Since 2009, Dr. Winkler is professor at the Department for Medical and Bioinformatics at the Upper Austria University of Applied Sciences, Campus Hagenberg.

ANDREAS BEHAM received his MSc in computer science in 2007 from Johannes Kepler University (JKU) Linz, Austria. His research interests include heuristic optimization methods and simulation-based as well as combinatorial optimization. Currently he is a research associate at the Research Center Hagenberg of the Upper Austria University of Applied Sciences (Campus Hagenberg).

MONIKA KOFLER studied Medical Software Engineering at the Upper Austrian University of Applied Sciences, Campus Hagenberg, Austria, from which she received her diploma's degree in 2006. She is currently employed as a research associate at the Research Center Hagenberg and pursues her PhD in engineering sciences at the Johannes Kepler University Linz, Austria.