Report from Dagstuhl Seminar 16412

Automated Algorithm Selection and Configuration

Edited by

Holger H. Hoos[1], Frank Neumann[2], and Heike Trautmann[3]

1 University of British Columbia, CA, hoos@cs.ubc.ca
2 University of Adelaide, AU, frank.neumann@adelaide.edu.au
3 Universität Münster, DE, trautmann@wi.uni-muenster.de

Abstract

This report documents the programme and the outcomes of Dagstuhl Seminar 16412 "Automated Algorithm Selection and Configuration", which was held October 9–14, 2016 and attended by 34 experts from 10 countries. Research on automated algorithm selection and configuration has lead to some of the most impressive successes within the broader area of empirical algorithmics, and has proven to be highly relevant to industrial applications. Specifically, high-performance algorithms for $\mathcal{NP}$-hard problems, such as propositional satisfiability (SAT) and mixed integer programming (MIP), are known to have a huge impact on sectors such as manufacturing, logistics, healthcare, finance, agriculture and energy systems, and algorithm selection and configuration techniques have been demonstrated to achieve substantial improvements in the performance of solvers for these problems. Apart from creating synergy through close interaction between the world's leading groups in the area, the seminar pursued two major goals: to promote and develop deeper understanding of the behaviour of algorithm selection and configuration techniques and to lay the groundwork for further improving their efficacy. Towards these ends, the organisation team brought together a group of carefully chosen researchers with strong expertise in computer science, statistics, mathematics, economics and engineering; a particular emphasis was placed on bringing together theorists, empiricists and experts from various application areas, with the goal of closing the gap between theory and practice.

Seminar October 9–14, 2016 – http://www.dagstuhl.de/16412
1998 ACM Subject Classification I.2 Artificial Intelligence, I.2.2 Automatic Programming, I.2.6 Learning, I.2.8 Problem Solving, Control Methods, and Search, G.1.6 Optimization
Keywords and phrases algorithm configuration, algorithm selection, features, machine learning, optimisation, performance prediction
Digital Object Identifier 10.4230/DagRep.6.10.33
Edited in cooperation with Marius Lindauer

1 Executive Summary

Holger H. Hoos
Frank Neumann
Heike Trautmann

The importance of high-performance algorithms, in particular for solving $\mathcal{NP}$-hard optimisation and decision problems, cannot be underestimated. Achievements in this area have substantial impact in sectors such as manufacturing, logistics, healthcare, finance, agriculture and energy systems – all of strategic importance to modern societies.

Automated Algorithm Selection and Configuration, *Dagstuhl Reports*, Vol. 6, Issue 10, pp. 33–74
Editors: Holger H. Hoos, Frank Neumann, and Heike Trautmann
Dagstuhl Reports
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

The development of effective automated algorithm selection and configuration techniques has been one of the major success stories in the area of empirical algorithmics in recent years. Building on a wide range of algorithmic approaches for problems such as propositional satisfiability (SAT) and mixed integer programming (MIP), these methods permit the selection of appropriate algorithms based on efficiently computable characteristic of a problem instance to be solved (algorithm selection) and the automatic determination of performance optimising parameter settings (algorithm configuration). In both cases, statistical models that enable performance predictions for previously unseen problem instances or parameter settings play a key enabling role; additionally, these models have other important uses, e.g., in load scheduling and distribution on large computer clusters.

The reach of those methods is illustrated by the fact that they have defined the state of the art in solving SAT, arguably the most prominent $\mathcal{NP}$-complete decision problem, for a decade (as witnessed by the results from the international SAT solver competitions (http://www.satcompetition.org), and more recently have been demonstrated to have the potential to achieve significant improvements over the long-standing state of the art in solving the TSP, one of the most widely studied $\mathcal{NP}$-hard optimisation problems [1]. Further very encouraging results have been achieved in recent years for continuous optimisation, AI planning and mixed integer programming problems.

The goal of the seminar was to foster research on algorithm selection and configuration, as well as on the underlying performance prediction methods, by bringing together researchers from the areas of artificial intelligence, theoretical computer science and machine learning in order to extend current studies to a much broader class of problems and build up the theoretical foundations of this important research area. On the foundational side, the seminar aimed at bridging the gap between experiments and theory in feature-based algorithm (runtime) analysis. In particular, we began investigating how mathematical and theoretical analyses can contribute to the experimentally driven research area of algorithm selection and configuration. We expect that studies following this initial exploration will bring together two of the currently most successful approaches for analysing heuristic search algorithms and ultimately achieve substantial impact in academic and industrial applications of algorithm configuration and selection techniques. Furthermore, we placed an emphasis on investigating automated algorithm selection and configuration approaches for multiobjective optimisation problems – an important, but largely unexplored area of investigation.

Background and Challenges: Algorithm Selection and Configuration for Combinatorial Problems

The design of algorithms for combinatorial optimisation and decision problems plays a key role in theoretical computer science as well as in applied algorithmics. These problems are frequently tackled using heuristic methods that perform extremely well on different classes of benchmark instances but usually do not have rigorous performance guarantees. Algorithm selection and configuration techniques have been applied to some of the most prominent $\mathcal{NP}$-hard combinatorial optimisation and decision problems, such as propositional satisfiability (SAT) and the travelling salesman problem (TSP).

Algorithm selection for SAT has first been explored in the seminal work on SATzilla [2, 3, 4], which was initially based on linear and ridge regression methods for performance prediction, but later moved to more sophisticated models based on cost-sensitive random forest classification [5]. Other successful methods use clustering techniques to identify the algorithm to be run on a given instance [6, 7]. As clearly evident from the results of SAT competitions, which are regularly held to assess and document the state of the art in SAT

solving, automated algorithm selection procedures effectively leverage the complementary strengths of different high-performance solvers and thus achieve substantial improvements over the best individual solvers [5].

As heuristic search algorithms often have numerous parameters that influence their performance, one of the classical questions is how to set parameters to optimise performance on a given class of instances. This per-set algorithm configuration problems can be solved using stochastic local search and model-based optimisation techniques [8, 9, 10], as well as racing techniques [11, 12], and these configuration methods have been demonstrated to yield substantial performance improvements to state-of-the-art algorithms for SAT, TSP, MIP, AI planning and several other problems (see, e.g., [13, 14, 15, 16]). Algorithm configuration techniques are now routinely used for optimising the empirical performance of solvers for a wide range of problems in artificial intelligence, operations research and many application areas (see, e.g., [17, 18]).

Initial work on combining algorithm selection and configuration techniques has shown significant promise [19, 20]; such combinations allow configuring algorithms on a per-instance basis [6, 7] and configuring algorithm selection methods (which themselves make use of many heuristic design choices) [21]. However, we see much room for further work along these lines. Other challenges concern the automated selection and configuration of mechanisms that adapt parameter settings while an algorithm is running and the configuration of algorithms for optimised scaling behaviour. Finally, a better theoretical foundation of algorithm selection and configuration approaches is desired and necessary. Initial steps into this direction were an important goal of this Dagstuhl seminar. In the following, we motivate and outline some of the challenges addressed in the course of the seminar.

Background and Challenges: Algorithm Selection for Continuous Black-Box Optimisation

Black-box function optimisation is a basic, yet intensely studied model for general optimisation tasks, where all optimisation parameters are real-valued. Work in this area has important practical applications in parameter and design optimisation and has also inspired some of the most successful general-purpose algorithm configuration techniques currently available [9].

Despite many years of research in metaheuristics, especially evolutionary algorithms, aimed at optimising black-box functions effectively, it is currently hardly possible to automatically determine a good optimisation algorithm for a given black-box function, even if some of its features are known. In single-objective (SO) black-box optimisation, it is therefore of considerable interest to derive rules for determining how problem properties influence algorithm performance as well as for grouping test problems into classes for which similar performance of the optimisation algorithms can be observed. Recent benchmarking experiments [22, 23] provide at best high-level guidelines for choosing a suitable algorithm type based on basic features that are known a priori, such as the number of dimensions of the given problem. However, the preference rules for algorithm selection thus obtained are very imprecise, and even for slight algorithm or problem variations, the resulting performance-induced ordering of different algorithms can change dramatically.

Exploratory Landscape Analysis (ELA, [24]) aims at improving this situation by deriving cheaply computable problem features based on which models relating features to algorithm performance can be constructed using benchmark experiments. The final goal is an accurate prediction of the best suited algorithm for an arbitrary optimisation problem based on the computed features. The concept is not entirely knew; however, earlier approaches, such as fitness distance correlation (FDC) [25], have not been completely convincing.

A first idea to employ high-level (human expert designed) features, such as separability and modality, to characterize optimisation problems in an ELA context [26] was therefore refined by also integrating low-level features – e.g., based on convexity or the behaviour of local search procedures[27]. These effectively computable low-level features can be chosen from a wide range of easy to measure statistical properties. Suitably determined combinations of such features are expected to provide sufficient information to enable successful algorithm selection. Following recent results [27], this process is not necessarily costly in terms of function evaluations required for feature computation.

Additional, conceptually similar features were introduced in [28, 29, 30, 31]. In [32], a representative portfolio of four optimisation algorithms was constructed from the complete list of BBOB 2009/2010 candidates. Based on the low-level features a sufficiently accurate prediction of the best suited algorithm within the portfolio for each function was achieved. Recently, the feature set was extended based on the cell mapping concept in [33] by which a finite subdivision of the domain in terms of hypercubes is constructed. Most recently, the ELA approach, extended by several specific features, was successfully used to experimentally detect funnel structured landscapes in unknown black-box optimisation problems [34]. As it can be assumed that this information can be efficiently exploited to speed up the optimisation process, we expect ELA to contribute importantly to automated algorithm selection in single-objective black-box optimisation. (See [35] for a survey of related work.)

One major challenge in this area is the construction of a suitable algorithm portfolio together with an algorithm selection mechanism for unknown instances that generalises well to practical applications. For this purpose, suitable benchmark sets have to be derived and the costs of feature computations have to be kept as small as possible. Furthermore, theoretical foundation of the approaches is desired and necessary. The seminar aimed to make first steps in this direction.

Special focus: Algorithm selection for multiobjective optimisation

Some of the most challenging real-world problems involve the systematic and simultaneous optimisation of multiple conflicting objective functions – for example, maximising product quality and manufacturing efficiency, while minimising production time and material waste. To solve such problems, a large number of multiobjective optimisation (MOO) algorithms has been reported. Like single-objective (SO) algorithms, new MOO algorithms are claimed to outperform others by comparing the results over a limited set of test problems. Knowles et al. [36] started working on systematically deriving performance measures for EMOA and evaluating EMOA performance. Mersmann et al. [37] recently derived a systematic benchmarking framework according to similar work of [38] on benchmarking classification algorithms.

However, it is unlikely that any algorithm would outperform all others on a broader set of problems, and it is possible that the algorithm fails miserably on some of them. These results go usually unreported, leaving the algorithm's limitations unknown. This knowledge is crucial to avoid deployment disasters, gain theoretical insights to improve algorithm design, and ensure that algorithm performance is robustly described. Therefore, we see much value in the development of an algorithm selection and configuration framework for multiobjective optimisation. Successfully selecting the proper optimization algorithm for a multi-objective problem depends on detecting different problem characteristics, one of which is the multimodality of the induced landscape. In recent work [39], formal definitions were introduced for multimodality in multi-objective optimization problems in order to generalize the ELA framework to multi-objective optimization.

Significant progress has been made on single-objective (SO) problems of combinatorial and continuous nature as discussed above. However, these ideas are yet to be applied to the important class of MOO problems. We see five major avenues of exploration: (1) analysis on what makes MOO problems difficult; (2) design of features to numerically characterize MOO problems; (3) identification and visualization of strengths and weaknesses of state-of-the-art MOO algorithms; (4) methodology to assist the algorithm selection and configuration on (possibly expensive) real-world problems; (5) methodology to assist the design of tailored algorithms for real-world problems. An important aim of the seminar was to facilitate discussion of these directions.

Seminar structure and outcomes

The seminar was structured to balance short invited presentations with group breakout sessions and a generous amount of time set aside for informal discussions and spontaneously organised working groups at a ratio of about 2:1:1. Based on feedback obtained during and after the event, this structure worked well in fostering a vibrant atmosphere of intense and fruitful exchange and discussion. Presenters very successfully introduced important ideas, outlined recent results and open challenges, and facilitated lively discussion that provided much additional value. The afternoon group breakout sessions were particularly effective in addressing the challenges previously outlined as well as additional topics of interest that emerged during the seminar – thanks to the preparation and moderation by the session organisers as well as the lively participation of the attendees.

While it would be unreasonable to expect to exhaustively or conclusively address the substantial research challenges that inspired us to organise this Dagstuhl seminar, we believe that very significant progress has been achieved. As importantly, we feel that through this week-long event, an invaluable sharing of perspective and ideas has taken place, whose beneficial effects on the algorithm selection and configuration community and its work we hope to be felt for years to come. The following presentation abstracts and session summaries provided by the participants reflect the richness and depth of the scientific exchange facilitated by the seminar.

As organisers, we very much enjoyed working with presenters and session organisers, who greatly contributed to the success of the seminar, as did everyone who participated. Our thanks also go to the local team at Schloss Dagstuhl, who provided outstanding organisational support and a uniquely inspiring environment.

References

1 L. Kotthoff, P. Kerschke, H. Hoos, and H. Trautmann. Improving the state of the art in inexact TSP solving using per-instance algorithm selection. In C. Dhaenens, L. Jourdan, and M.-E. Marmion, editors, *Learning and Intelligent Optimization, 9th International Conference*, pages 202–217, Cham, 2015. Springer International Publishing. Publication status: Published.

2 E. Nudelman, K. Leyton-Brown, H. H Hoos, A. Devkar, and Y. Shoham. Understanding random SAT: Beyond the clauses-to-variables ratio. In *Principles and Practice of Constraint Programming–CP 2004*, pages 438–452. Springer Berlin Heidelberg, 2004.

3 E. Nudelman, K. Leyton-Brown, A. Devkar, Y. Shoham, and H. Hoos. Satzilla: An algorithm portfolio for SAT. *Solver description, SAT competition*, 2004, 2004.

4 L. Xu, F. Hutter, H. Hoos, and K. Leyton-Brown. SATzilla: Portfolio-based algorithm selection for SAT. *Journal of Artificial Intelligence Research*, 32:565–606, 2008.

5 L. Xu, F. Hutter, H.H. Hoos, and K. Leyton-Brown. Evaluating component solver contributions to portfolio-based algorithm selectors. In *Theory and Applications of Satisfiability Testing–SAT 2012*, pages 228–241. Springer Berlin Heidelberg, 2012.

6 S. Kadioglu, Y. Malitsky, M. Sellmann, and K. Tierney. Isac-instance-specific algorithm configuration. *ECAI*, 215:751–756, 2010.

7 Y. Malitsky, A. Sabharwal, H. Samulowitz, and M. Sellmann. Algorithm portfolios based on cost-sensitive hierarchical clustering. In *Proceedings of the Twenty-Third international joint conference on Artificial Intelligence*, pages 608–614. AAAI Press, 2013.

8 F. Hutter, H.H. Hoos, K. Leyton-Brown, and K.P. Murphy. An experimental investigation of model-based parameter optimisation: Spo and beyond. In *GECCO'09: Proceedings of the 11th Annual conference on Genetic and evolutionary computation*, pages 271–278, New York, NY, USA, 2009. ACM.

9 F. Hutter, H.H. Hoos, and K. Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In *Learning and Intelligent Optimization*, pages 507–523. Springer Berlin Heidelberg, 2011.

10 C. Ansótegui, M. Sellmann, and K. Tierney. A gender-based genetic algorithm for the automatic configuration of algorithms. *Principles and Practice of Constraint Programming-CP 2009*, pages 142–157, 2009.

11 M. Birattari, T. Stützle, L. Paquete, and K. Varrentrapp. A racing algorithm for configuring metaheuristics. In *GECCO '02: Proc. of the Genetic and Evolutionary Computation Conference*, pages 11–18, 2002.

12 M. Birattari, Z. Yuan, P. Balaprakash, and T. Stützle. F-race and iterated F-race: An overview. In T. Bartz-Beielstein, M. Chiarandini, L. Paquete, and M. Preuss, editors, *Empirical Methods for the Analysis of Optimization Algorithms*. Springer, 2010.

13 F. Hutter, M.T. Lindauer, A. Balint, S. Bayless, H.H. Hoos, and K. Leyton-Brown. The configurable SAT solver challenge (CSSC). *Artificial Intelligence*, Accepted for publication., 2015.

14 J. Styles and H. Hoos. Ordered racing protocols for automatically configuring algorithms for scaling performance. In *Proceedings of the 15th Conference on Genetic and Evolutionary Computation (GECCO-13)*, pages 551–558. ACM, 2013.

15 F. Hutter, H.H. Hoos, and K. Leyton-Brown. Automated configuration of mixed integer programming solvers. In *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, pages 186–202. Springer Berlin Heidelberg, 2010.

16 M. Vallati, C. Fawcett, A.E. Gerevini, H.H. Hoos, and A. Saetti. Automatic generation of efficient domain-optimized planners from generic parametrized planners. In *Sixth Annual Symposium on Combinatorial Search (SoCS-13)*, pages 184–192, 2013.

17 P. Lengauer and H. Mössenböck. The taming of the shrew: Increasing performance by automatic parameter tuning for java garbage collectors. In *Proceedings of the 5th ACM/SPEC International Conference on Performance Engineering*, ICPE '14, pages 111–122, New York, NY, USA, 2014. ACM.

18 J.P. Dickerson, A.D. Procaccia, and T. Sandholm. Dynamic matching via weighted myopia with application to kidney exchange. In *Proceedings of the 28th National Conference on Artificial Intelligence (AAAI-14)*, pages 1340–1346, 2012.

19 L. Xu, H.H. Hoos, and K. Leyton-Brown. Hydra: Automatically configuring algorithms for portfolio-based selection. *Proceedings of the 24th Conference on Artificial Intelligence (AAAI-10)*, 10:210–216, 2010.

20 L. Xu, F. Hutter, H.H. Hoos, and K. Leyton-Brown. Hydra-MIP: Automated algorithm configuration and selection for mixed integer programming. *RCRA workshop on experimental evaluation of algorithms for solving problems with combinatorial explosion at the international joint conference on artificial intelligence (IJCAI)*, pages 16–30, 2011.

21 M. Lindauer, H. Hoos, F. Hutter and T. Schaub: AutoFolio: An Automatically Configured Algorithm Selector. J. Artif. Intell. Res. (JAIR) 53:745-778, 2015.

22 N. Hansen, A. Auger, S. Finck, and R. Ros. Real-parameter black-box optimization benchmarking 2009: Experimental setup. Technical Report RR-6828, INRIA, 2009.

23 N. Hansen, A. Auger, S. Finck, and R. Ros. Real-parameter black-box optimization benchmarking 2010: Experimental setup. Technical Report RR-7215, INRIA, 2010.

24 O. Mersmann, B. Bischl, H. Trautmann, M. Preuss, C. Weihs, and G. Rudolph. Exploratory landscape analysis. In *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, GECCO '11, pages 829–836, New York, NY, USA, 2011. ACM.

25 T. Jones and S. Forrest. Fitness distance correlation as a measure of problem difficulty for genetic algorithms. In *Proceedings of the Sixth International Conference on Genetic Algorithms*, pages 184–192. Morgan Kaufmann, 1995.

26 M. Preuss and Th. Bartz-Beielstein. Experimental analysis of optimization algorithms: Tuning and beyond. In Y. Borenstein and A. Moraglio, editors, *Theory and Principled Methods for Designing Metaheuristics*. Springer, 2011.

27 O. Mersmann, M. Preuss, H. Trautmann, B. Bischl, and C. Weihs. Analyzing the BBOB results by means of benchmarking concepts. *Evolutionary Computation Journal*, 23(1):161–185, 2015.

28 M. A. Muñoz, M. Kirley, and S. K. Halgamuge. A meta-learning prediction model of algorithm performance for continuous optimization problems. In C. A. Coello Coello et al., editors, *Parallel Problem Solving from Nature – PPSN XII*, volume 7491 of *Lecture Notes in Computer Science*, pages 226–235. Springer, 2012.

29 T. Abell, Y. Malitsky, and K. Tierney. Features for exploiting black-box optimization problem structure. In Giuseppe Nicosia and Panos Pardalos, editors, *Learning and Intelligent Optimization*, Lecture Notes in Computer Science, pages 30–36. Springer, 2013.

30 R. Morgan and M. Gallagher. Using landscape topology to compare continuous metaheuristics: A framework and case study on EDAs and ridge structure. *Evolutionary Computation*, 20(2):277–299, 2012.

31 M.A. Munoz, M. Kirley, and S.K. Halgamuge. Exploratory landscape analysis of continuous space optimization problems using information content. *Evolutionary Computation, IEEE Transactions on*, 19(1):74–87, Feb 2015.

32 B. Bischl, O. Mersmann, H. Trautmann, and M. Preuss. Algorithm selection based on exploratory landscape analysis and cost-sensitive learning. In *Proceedings of the 14th Annual Conference on Genetic and Evolutionary Computation*, GECCO '12, pages 313–320. ACM, 2012.

33 P. Kerschke, M. Preuss, C. Hernández, O. Schütze, J. Sun, C. Grimme, G. Rudolph, B. Bischl, and H. Trautmann. Cell mapping techniques for exploratory landscape analysis. In A Tantar et al., editors, *EVOLVE — A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation V*, volume 288 of *Advances in Intelligent Systems and Computing*, pages 115–131. Springer, 2014.

34 P. Kerschke, M. Preuss, S. Wessing, and H. Trautmann. Detecting funnel structures by means of exploratory landscape analysis. In *Proceedings of the*, pages 265–272, New York, NY, USA, 2015. ACM. Publication status: Published.

35 M. A. Muñoz, Y. Sun, M. Kirley, and S. K. Halgamuge. Algorithm selection for black-box continuous optimization problems: A survey on methods and challenges. *Information Sciences*, 317:224 – 245, 2015.

36 J. Knowles, L. Thiele, and E. Zitzler. A Tutorial on the Performance Assessment of Stochastic Multiobjective Optimizers. TIK Report 214, Computer Engineering and Networks Laboratory, ETH Zurich, 2006.

37 O. Mersmann, H. Trautmann, B. Naujoks, and C. Weihs. Benchmarking evolutionary multiobjective optimization algorithms. In *IEEE Congress on Evolutionary Computation*, pages 1–8. IEEE, 2010.

38 K. Hornik and D. Meyer. Deriving consensus rankings from benchmarking experiments. In R. Decker and H.-J. Lenz, editors, *Advances in Data Analysis (Proc. of the 30th Ann. Conf. of the Gesellschaft für Klassifikation*, pages 163–170. Springer, Berlin, 2007.

39 P. Kerschke, H. Wang, M. Preuss, C. Grimme, A. Deutz, H. Trautmann and M. Emmerich. Towards Analyzing Multimodality of Multiobjective Landscapes. In *Proceedings of the 14^{th} International Conference on Parallel Problem Solving from Nature (PPSN XIV)*, pages 962–972. Lecture Notes in Computer Science, Springer, 2016.

2 Table of Contents

3 Presentations

3.1 Continuous Black-Box Optimization: How Large Are The GAPs?

Anne Auger (INRIA Saclay – Orsay, FR)

This talk was motivated by the breakout session on Theory versus Practice or "All you ever wanted to know/ask a theoretician and All you ever wanted to know/ask a practitioner" where the observation was made that gaps between theory and practice is larger or smaller depending on the domain and community.

We have discussed for the domain of continuous black-box optimization or adaptive stochastic search algorithms where theory stands with respect to practice. In particular we have sketched how theoretical results on linear convergence relate to practice and how they are motivated by practice. We have also highlighted a few lessons from theory to practice. Last we have discussed this gap between communities tackling the same problem namely the Derivative Free Optimization community and the Evolutionary Computation community and how there are signs that this gap in becoming narrower and narrower.

3.2 Tuning using Multiple Criteria – Forwarding more Information to the Configurator

Aymeric Blot (INRIA Lille, FR)

Joint work of Aymeric Blot, Holger Hoos, Marie-Éléonore Kessaci-Marmion, Laetitia Jourdan, Heike Trautmann

In automatic algorithm configuration, a single performance indicator of the target algorithm is usually forwarded to the configurator. We discuss problems and possible solutions in cases where more that a single indicator might be needed. The highlighted solution is MO-ParamILS, a configurator specifically designed for the purpose of performing the configuration process using Pareto dominance on multiple performance indicators.

3.3 Some Ideas on Industrial Applications of Automated Optimizer Design

Thomas Bäck (Leiden University, NL)

Joint work of Sander van Rijn, Hao Wang, Matthijs van Leeuwen, Thomas Bäck
Main reference S. van Rijn, H. Wang, M. van Leeuwen, T. Bäck, "Evolving the Structure of Evolution Strategies", arXiv:1610.05231v1 [cs.NE], 2016.
URL https://arxiv.org/abs/1610.05231v1

Many industrial applications are represented by simulation models which require enormous computational effort for computing a single objective function value. Motivated by such applications, e.g., in the automotive industry, an important requirement for optimization

algorithms can be to deliver large improvements with the smallest number of function evaluations possible.

From an automated optimizer design perspective, I discuss some preliminary experiments on the automatic configuration of algorithmic variants of modern evolutionary strategies. As these results illustrate, it is possible to significantly improve performance by configuring the components of evolutionary strategies optimally.

The talk then presents my vision on how to extend this approach towards automated optimizer design for simulation-based function classes. This might involve response surface modeling, exploratory feature analysis, machine learning, and aspects of genetic programming and grammatical evolution – plus likely a number of additional techniques for this challenging problem.

3.4 Learning the Right Mutation Strength on-the-Fly

Benjamin Doerr (Ecole Polytechnique – Palaiseau, FR)

Joint work of Benjamin Doerr, Carola Doerr, Jing Yang

When using the classic standard bit mutation operator, parent and offspring differ in a random number of bits, distributed according to a binomial law. This has the advantage that all Hamming distances occur with some positive probability, hence this operator can be used, in principle, for all fitness landscapes. The downside of this "one-size-fits-all" approach, naturally, is a performance loss caused by the fact that often not the ideal number of bits is flipped. Still, the fear of getting stuck in local optima has made standard bit mutation become the preferred mutation operator.

In this work we show that a self-adjusting choice of the number of bits to be flipped can both avoid the performance loss of standard bit mutation and avoid the risk of getting stuck in local optima. We propose a simple mechanism to adaptively learn the currently optimal mutation strength from previous iterations. This aims both at exploiting that generally different problems may need different mutation strengths and that for a fixed problem different strengths may become optimal in different stages of the optimization process.

We experimentally show that our simple hill climber with this adaptive mutation strength outperforms both the randomized local search heuristic and the (1+1) evolutionary algorithm on the LeadingOnes function and on the minimum spanning tree problem. We show via mathematical means that our algorithm is able to detect precisely (apart from lower order terms) the complicated optimal fitness-dependent mutation strength recently discovered for the OneMax function. With its self-adjusting mutation strength it thus attains the same runtime (apart from $o(n)$ lower-order terms) and the same (asymptotic) 13% fitness-distance improvement over RLS that was recently obtained by manually computing the optimal fitness-dependent mutation strength.

This talk is based on joint work with Carola Doerr (Paris 6) and Jing Yang (École Polytechnique)

References

1 Benjamin Doerr, Carola Doerr, Jing Yang. k-Bit Mutation with Self-Adjusting k Outperforms Standard Bit Mutation. In *Parallel Problem Solving from Nature – PPSN XIV*, pages 824–834, 2016.

3.5 Self-Adjusting Choice of Population Size and Mutation Strengths in Discrete Optimization

Carola Doerr (CNRS and University Pierre & Marie Curie – Paris, FR)

In most evolutionary algorithms there are a number of parameters to be chosen, e.g., the population size, the mutation strength, the crossover rate, etc. While it seems quite intuitive that different parameter choices can be optimal in the different stages of the optimization process, little theoretical evidence exist to support this claim for discrete optimization problems. In two recent works [Doerr/Doerr, Optimal Parameter Choices Through Self-Adjustment: Applying the 1/5-th Rule in Discrete Settings, GECCO 2015] and [Doerr/Doerr/Kötzing: Provably Optimal Self-adjusting Step Sizes for Multi-valued Decision Variables, PPSN 2016] we propose a simple success-based update rule for the population size and the mutation strength, respectively. In both these works we show that the self-adjusting parameter choice yields a better performance than any (!) static parameter choice. The update rule is inspired by the classical one-fifth rule from continuous optimization. Based on joint work with Benjamin Doerr (Ecole Polytechnique, France) and Timo Koetzing (HPI Potsdam, Germany)

3.6 Spotlight on Rumor Sreading

Carola Doerr (CNRS and University Pierre & Marie Curie – Paris, FR)

In this short talk we briefly discuss the rumor spreading problem and the main objectives that we are after when designing algorithms for it.

Rumor spreading aims at distributing information in networks via so-called PUSH operations. Informed nodes are allowed to call others to inform them. We aim at protocols that are fast, need few calls, are robust with respect to node and edge crashes, and hopefully simple. Existing works analyze rumor spreading algorithms on different types of graphs, e.g., social networks and dynamically changing graphs.

3.7 Algorithm Subset Selection as a Portfolio Investment Problem

Michael Emmerich (Leiden University, NL)

Joint work of Michael Emmerich, Iryna Yevseyeva

The problem of optimization algorithm selection (and configuration) can be formulated in a similar way than a financial portfolio investment problem with risk and expected return. Depending on the practical setting in which the optimization algortihms are applied, different scenarios and problem formulations can be of interest.

Here, as a pars pro toto for a larger class of problem formulations, one particular scenario is discussed. The chosen example formulation is motivated by problems that occur in logistics planning, and in real-world environments of computer-aided product design, and it reads as

follows: Prior to performing an time-expensive optimization taks, a single algorithm or a subset of algorithms has to be selected from a library or algorithm portfolio. The algorithm (or the subset of algorithms) are then submitted to a parallel computation cluster where they have to solve an a priori unknown problem instance. After a pre-assigned time the results achieved by all selected algorithms are collected and the best result is chosen.

This scenario leads to a theoretical problem formulation where each algorithm is viewed as an investment (asset) and its result, which has to be maximized, is viewed as the financial return of this investment. Due to the uncertainty about the particular problem instance that will have to be solved, the return will be considered as a random variable. Given a single algorithm the expected value of the return has to be maximized and the risk, which is related to the variance of the return, is to be minimized. For a subset of algorithms, the expected maximum of the return has to be maximized, and the risk related to this value needs to be minimized. The computation of the risk term requires covariances of the returns of the algorithms, noting that diversification will usually be beneficial for reducing the risk.

There is a rich theory on the solution of such multiobjective portfolio selection problems, which can be transferred to the algorithm selection domain. However, there are also challenges to be overcome, such as the elicitation or estimation of probability distributions and the approximation or computation of the efficient set when it comes to large algorithm libraries or instance spaces, as they would need to be considerd in algorithm tuning or configuration.

3.8 Optimisation Algorithm Design: A Control Engineering Perspective

Carlos M. Fonseca (University of Coimbra, PT)

Joint work of Cláudia R. Correa, Elizabeth F. Wanner, Carlos M. Fonseca, Rodrigo T. N. Cardoso, Ricardo H. C. Takahashi
Main reference C. R. Correa, E. F. Wanner, and C. M. Fonseca, "Lyapunov design of a simple step-size adaptation strategy based on success", in Proc. of the 14th Int'l Conf. on Parallel Problem Solving from Nature – PPSN XIV, LNCS, Vol. 9921, pp. 101–110, Springer, 2016.
URL http://dx.doi.org/10.1007/978-3-319-45823-6_10

Currently, most practically-relevant metaheuristic algorithms, and Evolutionary Algorithms (EAs) in particular, are not amenable to analysis with the available theoretical tools. On the other hand, EA theory has focused largely on asymptotic and time-complexity results in ideal or much simplified scenarios, which are not immediately useful to practitioners.

An alternative route for theoretical development is to approach the design of such optimisation algorithms from a control engineering perspective, where determining algorithm parameters is *the* purpose of the analysis, and algorithms must be designed with analysis in mind. The fact that optimisation algorithms are inherently dynamical systems further substantiates this point of view. Moreover, theory should support the use of numerical methods to extend the analysis to larger and/or more complex scenarios before experimentation becomes the only practical alternative.

This perspective will be illustrated with the design of a simple step-size adaptation strategy based on success and failure events [1]. A Lyapunov synthesis procedure is used to obtain both a performance guarantee and tuned adaptation parameter values. The method relies on the numerical optimisation of an analytically-derived performance index.

Acknowledgement. This work was partially supported by national funds through the Portuguese Foundation for Science and Technology (FCT) and by the European Regional

Development Fund (FEDER) through COMPETE 2020 – Operational Program for Competitiveness and Internationalisation (POCI).

References

1 C. R. Correa, E. F. Wanner, C. M. Fonseca, "Lyapunov design of a simple step-size adaptation strategy based on success," in Proc. of the 14th Int'l Conf. on Parallel Problem Solving from Nature – PPSN XIV, LNCS, Vol. 9921, pp. 101–110, Springer, 2016.

3.9 Models of Large Real-world Networks

Tobias Friedrich (Hasso-Plattner-Institut – Potsdam, DE)

The node degrees of large real-world networks often follow a power-law distribution. Such scale-free networks can be social networks, internet topologies, the web graph, power grids, or many other networks from literally hundreds of domains. The talk introduced several mathematical models of scale-free networks (e.g. preferential attachment graphs, Chung-Lu graphs, hyperbolic random graphs), showed some of their properties (e.g. diameter, average distance, clustering), and discussed how these properties influence algorithm selection and configuration.

3.10 Needed: Hard and Killer problems

Marcus Gallagher (The University of Queensland – Brisbane, AU)

In this short talk, I would like to raise issues around the nature of optimization problems. What sorts of benchmark problems are needed to extract maximum value from our experiments? Do we need hard problems (but with rich structure) to solve and where are they (especially in the continuous case)? Finally, what are the so-called "killer apps" for the field?

3.11 Fixed-Parameter Single Objective Search Heuristics for Minimum Vertex Cover

Wanru Gao (University of Adelaide, AU)

Joint work of Wanru Gao, Tobias Friedrich, Frank Neumann
Main reference W. Gao, T. Friedrich, F. Neumann, "Fixed-Parameter Single Objective Search Heuristics for Minimum Vertex Cover", in Proc. of the 14th Int'l Conf. on Parallel Problem Solving from Nature – PPSN XIV, LNCS, Vol. 9921, pp. 740–750, Springer, 2016.
URL http://dx.doi.org/10.1007/978-3-319-45823-6_69

We consider how well-known branching approaches for the classical minimum vertex cover problem can be turned into randomized initialization strategies with provable performance guarantees and investigate them by experimental investigations. Furthermore, we show

how these techniques can be built into local search components and analyze a basic local search variant that is similar to a state-of-the-art approach called NuMVC. Our experimental results for the two local search approaches show that making use of more complex branching strategies in the local search component can lead to better results on various benchmark graphs.

3.12 AutoML – with a focus on deep learning

Frank Hutter (Universität Freiburg, DE)

The rapid growth of machine learning (ML) applications has created a demand for off-the-shelf ML methods that can be used easily and without expert knowledge. I first briefly review the successful approach of casting this problem as an optimization problem on top of a highly-parameterized ML framework. Then, I focus on possible extensions of this approach that could scale it up to achieve fully automated deep learning: reasoning across datasets, subsets of data, and initial time steps; online hyperparameter control; and automatically deriving insights. Many of these extensions have a direct correspondence in optimizing hard combinatorial problem solvers.

3.13 Applications in Hospital

Laetitia Jourdan (INRIA Lille, FR)

Joint work of Clarisse Dhaenens, Laetitia Jourdan
Main reference J. Jacques, J. Taillard, D. Delerue, C. Dhaenens, L. Jourdan, "Conception of a dominance-based multiobjective local search in the context of classification rule mining in large and imbalanced data sets", Applied Soft Computing, Vol. 34, pp. 705–720, Elsevier, 2015.
URL http://dx.doi.org/10.1016/j.asoc.2015.06.002
Main reference K. Seridi, L. Jourdan, E.-G. Talbi, "Using multiobjective optimization for biclustering microarray data", Applied Soft Computing, Vol. 33, pp. 239–249, 2015.
URL http://dx.doi.org/10.1016/j.asoc.2015.03.060
Main reference J. Hamon, J. Jacques, L. Jourdan, C. Dhaenens, "Knowledge Discovery in Bioinformatics", in Handbook of Computational Intelligence, pp. 1211–1223, Springer, 2015.
URL http://dx.doi.org/10.1007/978-3-662-43505-2_61
Main reference J. Jacques, J. Taillard, D. Delerue, L. Jourdan, C. Dhaenens, "The benefits of using multi-objectivization for mining pittsburgh partial classification rules in imbalanced and discrete data", in Proc. of the 15th Annual Conf. on Genetic and Evolutionary Computation (GECCO 2013), pp. 543–550, ACM, 2013.
URL https://dx.doi.org/10.1145/2463372.2463432

Hospital offers a lot of real world problems for the optimization and machine learning community. A lot of classical operation research problems can be found but often with a lot of additional constraints, presence of uncertainty and even dynamism of data. Concerning machine learning problems, there are very specific like classification on imbalanced data, bi-clustering and often solvers are not available in classical framework or they cannot cope the dimension of the data. All these problems can be modelled as optimisation problems often multiobjective problems. As final output should be software for non-domain specialists, automated configuration is required BUT how to realized it when there is only one dataset available, dataset that is often available only inside the hospital. Additionally, the robustness

of the proposed solutions is very important, as it can be critical for the hospital, how practitioners can assess the sensitivity of the found algorithms ?

3.14 Extending Exploratory Landscape Analysis Toward multiobjective and Multimodal Problems

Pascal Kerschke (Universität Münster, DE) and Mike Preuß (Universität Münster, DE)

Main reference O. Mersmann, B. Bischl, H. Trautmann, M. Preuß, C. Weihs, G. Rudolph, "Exploratory landscape analysis", in Proc. of the 13th Annual Conf. on Genetic and Evolutionary Computation (GECCO 2011), pp. 829–836, ACM, 2011.
URL https://dx.doi.org/10.1145/2001576.2001690
Main reference P. Kerschke, M. Preuß, S. Wessing, H. Trautmann, "Low-Budget Exploratory Landscape Analysis on Multiple Peaks Models", in Proc. of the 18th Annual Conf. on Genetic and Evolutionary Computation (GECCO 2016), pp. 229–236, ACM, 2016.
URL https://dx.doi.org/10.1145/2908812.2908845
Main reference P. Kerschke, H. Wang, M. Preuß, C. Grimme, A. H. Deutz, H. Trautmann, M. Emmerich, "Towards Analyzing Multimodality of Continuous Multiobjective Landscapes", in Proc. of the 14th Int'l Conf. on Parallel Problem Solving from Nature – PPSN XIV, LNCS, Vol. 9921, pp. 962–972, Springer, 2016.
URL http://dx.doi.org/10.1007/978-3-319-45823-6_90

Selecting the best suited algorithm for an optimization problem is usually a complex and difficult task, especially for expensive Black-Box problems. The Exploratory Landscape Analysis (ELA) approach extracts – not necessarily intuitively understandable – landscape features based on a (usually rather small) initial sample of observations from the underlying optimization problem. In case of population based algorithms, a well distributed initial population may be used as sample data for computing these features, which may then be used to enhance the algorithm selection model. So far, ELA is mostly used in the context of continuous, single-objective, global optimization problems – but it shall be transferred also to other domains. Next to a minimal introduction and report on the current state of ELA, we highlight the possibilities to extend it onto multiobjective and multimodal optimization.

3.15 Deep Parameter Configuration (joint work with UCL)

Lars Kotthoff (University of British Columbia – Vancouver, CA)

Algorithm configuration has been limited to parameters that the programmer intentionally exposes. Automatic algorithm configuration makes it feasible to efficiently explore ever larger parameter spaces, but the mindset of programmers is still that parameters should be exposed sparingly as they put additional burden on the user. We leverage techniques from software engineering to expose additional parameters from the source of an algorithm, thus increasing the potential gains for algorithm configuration.

3.16 Algorithm Portfolios: Four Key Questions

Kevin Leyton-Brown (University of British Columbia – Vancouver, CA)

This talk considered four key questions:

1. How useful is my solver? I argued this is well answered using the Shapley value.
2. How useful is my algorithm selector? I argued that one needs to be careful to avoid using a biased estimate of VBS performance.
3. How useful are my selector's features? I argued that just size-based features are often enought.
4. How useful is my benchmark? I argued that heterogeneity may produce artificially easy test data.

3.17 Multiobjective Optimization from the Perspective of Game Theory

Kevin Leyton-Brown (University of British Columbia – Vancouver, CA)

I described what multiobjective optimization means to a game theorist. I discussed von Neumann-Morgenstern utility theory, multiattribute utility, noncooperative game theory. I discussed the solution concepts Pareto optimality, stability concepts like Nash equilibrium, robustness concepts like maxmin, and minimax regret.

3.18 Combining Algorithm Selection and Configuration: Per-Instance Algorithm Configuration

Marius Lindauer (Universität Freiburg, DE)

Algorithm configuration and algorithm selection perform well in different use cases, namely homogeneous instance sets with large parameter configuration spaces vs heterogeneous instances with a small finite portfolio of algorithms. One way to combine algorithm selection and configuration is for example to apply configuration on top of selection [1]. However, to deal with heterogeneous instances (e.g., hard combinatorial problems, machine learning data sets or environmental variables) and an algorithm with a large parameter configuration space, we need a direct combination of configuration and selection:

Per-Instance algorithm configuration (PIAC) approaches (such as ISAC [2] and Hydra [3]) were proposed already some years ago. In the meantime, relevant techniques for PIAC made substantial progress, e.g., prediction of performance [4], quantification of homogeneity [5] and feature-parameters mappings [6]. Using recent advances in these areas, I believe that we can do much better than previous approaches in generating robust algorithms that adapt their parameter settings on a per-instance base.

References

1 Marius Lindauer, Holger H. Hoos, Frank Hutter, Torsten Schaub: *AutoFolio: An Automatically Configured Algorithm Selector*. J. Artif. Intell. Res. (JAIR) 53:745-778 (2015)
2 Carlos Ansotegui, Joel Gabas, Yuri Malitsky, Meinolf Sellmann: *MaxSAT by improved instance-specific algorithm configuration.* Artif. Intell. 235: 26-39 (2016)
3 Lin Xu, Holger Hoos, Kevin Leyton-Brown: *Hydra: Automatically Configuring Algorithms for Portfolio-Based Selection.* AAAI 2010
4 Frank Hutter, Lin Xu, Holger H. Hoos, Kevin Leyton-Brown: *Algorithm runtime prediction: Methods and evaluation.* Artif. Intell. 206: 79-111 (2014)
5 Marius Schneider, Holger H. Hoos: *Quantifying Homogeneity of Instance Sets for Algorithm Configuration.* LION 2012: 190-204
6 Jakob Bossek, Bernd Bischl, Tobias Wagner, Günter Rudolph: *Learning Feature-Parameter Mappings for Parameter Tuning via the Profile Expected Improvement.* GECCO 2015: 1319-1326

3.19 Network on Selection and Configuration: COSEAL

Marius Lindauer (Universität Freiburg, DE)

Since algorithm selection and algorithm configuration is widely applicable in many domains (including e.g., machine learning, hard combinatorial problems and continuous optimization), there are sub-communities in all these domains that use automatic selection and configuration of algorithms to improve the performance of their algorithms. Unfortunately, these communities were not well connected, even though they worked on similar problems. To change this, the COSEAL group (COnfiguration and SEection of ALgorithms)[1] was founded three years ago to create a research network on automatic selection and configuration of any kind of algorithm.

To encourage exchange of progress and expert knowledge between the different communities, the COSEAL group has an active mailing list and an annual workshop meeting. The mailing list is an open platform where everyone can join. Its intended use includes the announcement of new important results, tools, and to request help for newcomers. Similar to our Dagstuhl seminar, the goal of the workshops is less to promote newly published papers but to discuss on-going projects and open questions. To this end, the workshop includes sessions for presentations, posters and discussions. One of the successful projects of the COSEAL group was also launched at the first COSEAL workshop: the creation of a benchmark library for algorithm selection, called ASlib [1]. Furthermore, the COSEAL website offers overviews and literature links for algorithm selection and configuration for new researchers.

References

1 Bischl, B., Kerschke, P., Kotthoff, L., Lindauer, M., Malitsky, Y., Frechétte, A., Hoos, H., Hutter, F., Leyton-Brown, K., Tierney, K. and Vanschoren, J. ASlib: *A Benchmark Library for Algorithm Selection* In: Artificial Intelligence Journal (AIJ) 237 (2016): 41-58

[1] http://www.coseal.net

3.20 Challenges in Automated Algorithm Design: Representativeness, one-shot expensive scenarios, parameter importance and sensitivity, and human-in-the-loop

Manuel López-Ibánez (Univ. of Manchester, GB)

When facing realistic scenarios of automatic algorithm configuration and design, there are some questions for which our current answers appear lacking to practitioners. One practical question is how to create a set of instances representative of a problem or, alternatively, if only a small set of such instances is available, how to split the set between training and validation making sure that the training set remains representative of the target problem. Moreover, in some scenarios, we may be interested in solving a single very expensive problem instance once with the best algorithm possible. What are the strategies that automated algorithm configuration can offer in such scenarios? Another frequent question from practitioners is how to evaluate our confidence in the best configurations found, how important are the settings chosen and how sensitive are these particular settings. Some work has been done in this regard, but there are still many open questions. Finally, a more recent question is how to best integrate human interaction in the automated configuration procedure, when the target algorithm relies on a human to guide them to the optimal solution, such as in multi-criteria optimization methods that elicit preferences from decision-makers.

3.21 Building and exploiting a non-parametric problem space

Andres Munoz Acosta (Monash University – Clayton, AU)

While automated algorithm selection methods have been very successful in practice, in some cases they depend on features that can be qualified as heuristics. Therefore, there are no guarantees that the representation of the problem space is efficient; or that the insights gained from the features can be turned into useful algorithms. In other words, how to exploit the features to construct useful algorithms? Perhaps the first step is to produce an efficient map of the problem space, such that the map is one-to-one. Then, we may be able to identify a transformation that would convert a new problem into a previously observed one for which we know a good –or the best– algorithm. While similar ideas exist in the literature, e.g., merging branches and nodes to make a coarse grained version of the problem, such transformations are somewhat generic. On a side note, can we find such transformations in a principled and efficient way, perhaps on-line?

References

1 M.A. Munoz and K.A. Smith-Miles, “Performance analysis of continuous black-box optimization algorithms via footprints in instance space”, Evol. Comput., http://dx.doi.org/10.1162/EVCO_a_00194.

3.22 Automated Selection of Tree Decompositions

Nysret Musliu (TU Wien, AT)

Joint work of Michael Abseher, Frederico Dusberger, Nysret Musliu, Stefan Woltran
Main reference M. Abseher, F. Dusberger, N. Musliu, S. Woltran, "Improving the Efficiency of Dynamic Programming on Tree Decompositions via Machine Learning", in Proc. of the 24th Int'l Joint Conf. on Artificial Intelligence (IJCAI 2015), pp. 275–282, AAAI Press/IJCAI, 2015.
URL http://ijcai.org/Abstract/15/045

Dynamic Programming (DP) over tree decompositions is a well-established method to solve problems that are in general NP-hard – efficiently for instances of small treewidth. Experience shows that DP algorithms exhibit a high variance in runtime when using different tree decompositions (TD). In fact, given an instance of the problem at hand, even decompositions of the same width might yield extremely diverging runtimes.

We propose a general method that is based on selection of the best decomposition from an available pool of heuristically generated ones. Novel features for tree decomposition are proposed and machine learning techniques are applied to select the most promising decomposition. Extensive experiments in different problem domains show a significant speedup when choosing the tree decomposition according to this concept over simply using an arbitrary one of the same width.

3.23 Feature-Based Diversity Optimization for Problem Instance Classification

Samadhi Nethmini Nallaperuma (University of Sheffield, GB)

Joint work of Wanru Gao, Samadhi Nethmini Nallaperuma, Frank Neumann

Understanding the behaviour of heuristic search methods is a challenge. This even holds for simple local search methods such as 2OPT for the Traveling Salesperson problem. In this paper, we present a general framework that is able to construct a diverse set of instances that are hard or easy for a given search heuristic. Such a diverse set is obtained by using an evolutionary algorithm for constructing hard or easy instances that are diverse with respect to different features of the underlying problem. Examining the constructed instance sets, we show that many combinations of two or three features give a good classification of the TSP instances in terms of whether they are hard to be solved by 2OPT.

This research has been supported by the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 618091 (SAGE) and by the Australian Research Council under grant agreement DP140103400.

3.24 Algorithm Selection in Music Data Analysis

Günter Rudolph (TU Dortmund, DE)

Joint work of Günter Rudolph, Igor Vatolkin

The analysis of signal data arising from music recordings offers many ways to apply machine learning methods. In case of classification tasks many different algorithms may be deployed which must be configured appropriately. The manual selection and parameterization of algorithmic alternatives is often necessary in most systems that realize the classification chain of musical data. The automation of this task offers the perspective of huge improvements in performance. The next years will show how much can be gained by deep neural networks that are currently built into existing systems.

3.25 Cognitive Assistant for Data Scientist

Horst Samulowitz (IBM TJ Watson Research Center – Yorktown Heights, US)

Joint work of Gregory Bramble, Maria Butrico, Andre Cunha, Elias Khalil, Udayan Khurana, Peter Kirchner, Tim Klinger, Fatemeh Nargesian, Srinivasan Parthasarathy, Chandra Reddy, Anton Riabov, Horst Samulowitz, Gerry Tesauro, Deepak Turaga

A Data Scientist typically performs a number of tedious and time-consuming steps to derive insight from a raw data set. The process usually starts with data ingestion, cleaning, transformation (e.g. outlier removal, missing value imputation), then model building, and finally a presentation of predictions that align with the end-users objectives and preferences. It is a long, complex, and sometimes artful process requiring substantial time and effort especially because of the combinatorial explosion in choices of algorithms (and platforms), their parameters, and their compositions. Tools that can help automate steps in this process have the potential to accelerate the time-to-delivery of useful results, expand the reach of data science to non-experts, and offer a more systematic exploration of the available options.

This system aims at showing how automatic composition and configuration of data analytics (spanning multiple analytic platforms and packages such as R, Weka, SPSS, Apache SPARK, System ML) can offer increased insight into the data and how model selection algorithms can suggest models that are well suited to the predictive task, while respecting user preferences. Given a data set and analysis task (e.g., classification or regression) the system aims to quickly determine an appropriate combination of preprocessing steps (e.g., feature reduction or mapping) and models and platforms to achieve the users goals.

During this process the user is presented with intermediate results and insights into the data and the reasoning process itself. For example, which features are important? How well do entire classes of analytic tools perform on the data set? Are there potentially significant outliers? The user can directly interact with the process providing resource constraints (e.g., time available for training/testing) or their preference (e.g. for interpretable models). In addition, the system aims to provide basic suggestions of potentially related work that may enable the data scientist to improve results even further.

The system is constructed on top of an automated analytics composer and analytic repository which supports massively parallel execution and cross-platform execution of a wide range of high-performance analytic tools. To automatically select, compose and configure

these analytics we develop meta-learning algorithms that attempt to rapidly estimate upside performance using only subset of the available data. Furthermore, it tries to exploit structured as well as unstructured data to provide further suggestions.

3.26 From off-line to on-line feature-based parameter tuning

Marc Schoenauer (INRIA Saclay – Orsay, FR)

Main reference N. Belkhir, J. Dréo, P. Savéant, M. Schoenauer, "Feature Based Algorithm Configuration: A Case Study with Differential Evolution", in Proc. of the 14th Int'l Conf. on Parallel Problem Solving from Nature – PPSN XIV, LNCS, Vol. 9921, pp. 156–165, Springer, 2016.
URL http://dx.doi.org/10.1007/978-3-319-45823-6_15

Off-line parameter tuning based on feature computation can be achieved via the learning of an Empirical Performance Model trained on a large dataset of features x parameters, performance instances (involving huge computational cost, but this is not the point here). Given an unknown instance with computed features, optimal parameters according to the EPM can be derived. The features are of course problem-dependent, but quite often involve the computation of the objective values on a (as small as possible) set of sample points uniformly drawn from the design space – and hence can be thought of as global features. However, during the search itself, more points of the design space are sampled, and if the features are re-computed using this biased sample (or adding it to the original sample), some more "local" values of the features are obtained, that might lead to new optimal parameters according to the EPM. Very preliminary results have been obtained for continuous optimizaiton using DE (see PPSN 2016 paper by Belkir et al.).

3.27 Cognitive Computing

Meinolf Sellmann (IBM TJ Watson Research Center – Yorktown Heights, US)

The role of IT is fundamentally shifting as our ability to collect and harness ever growing amounts of data improves. Historically an enabler of business, IT is now moving closer and closer to the heart of modern economies by informing and influencing key strategic business decisions. Human data science labor cannot keep up with the ever growing demand for decision support analytic models. The programmable era is thus coming to an end as the cognitive era of machines that practice self-orientation begins. In this presentation I invite the participants to discuss the role of meta-algorithmics in the cognitive era.

3.28 Automated generation of high-performance heuristics from flexible algorithm frameworks: Challenges and Perspectives

Thomas Stützle (Free University of Brussels, BE)

The design of algorithms for computationally hard problems is time-consuming and difficult for a number of reasons such as the complexity of such problems, the large number of degrees of freedom in algorithm design and the setting of numerical parameters, and the difficulties of algorithm analysis due to heuristic biases and stochasticity. In recent years, automatic algorithm configuration methods have been developed to effectively search large and diverse parameter spaces; these methods have been shown to be able to identify superior algorithm designs and to find performance improving parameter settings.

In this talk, I will shortly summarize the main recent results that we have obtained in the automatic design of hybrid stochastic local search algorithms as well as multiobjective optimizers. We show that even for problems that have received very high attention in the literature new state-of-the-art algorithms can be obtained automatically, that is, without manual algorithm tuning. I will use these recent advances to discuss informally possible directions for the future work in this direction and discuss possible challenges.

3.29 Bringing the human back in the loop

Joaquin Vanschoren (TU Eindhoven, NL)

There has been great progress on fully-automated approaches for machine learning. Sometimes, however, human experience, intuition, or domain knowledge can prove valuable to constrain the space of solutions to explore. This is especially true when we consider the larger pipeline of data science, which also includes data wrangling, data cleaning, data integration, and model post-processing, among others. In this short talk, I would like to discuss and elicit ways to couple human expertise and machine learning to create a human-machine symbiosis. The human scientist would focus on the science (follow hunches, include more data,...) while letting the machine take care of drudge work (finding similar datasets, selecting algorithms/hyperparameters,...), thus enabling her to make informed, data-driven decisions. Meanwhile, the machine would learn from *all* the experiments run during these collaborations (with many people), and leverage what it learned from previous problems to help humans better in the future.

3.30 A Generic Bet-and-run Strategy for Speeding Up Traveling Salesperson and Minimum Vertex Cover

Markus Wagner (University of Adelaide, AU)

Joint work of Tobias Friedrich, Timo Kötzing, Markus Wagner
Main reference T. Friedrich, T. Kötzing, M. Wagner, "A Generic Bet-and-run Strategy for Speeding Up Traveling Salesperson and Minimum Vertex Cover", arXiv:1609.03993v1 [cs.AI], 2016.
URL https://arxiv.org/abs/1609.03993v1

A common strategy for improving optimization algorithms is to restart the algorithm when it is believed to be trapped in an inferior part of the search space. However, while specific restart strategies have been developed for specific problems (and specific algorithms), restarts are typically not regarded as a general tool to speed up an optimization algorithm. In fact, many optimization algorithms do not employ restarts at all.

Recently, bet-and-run was introduced in the context of mixed-integer programming, where first a number of short runs with randomized initial conditions is made, and then the most promising run of these is continued. In this article, we consider two classical NP-complete combinatorial optimization problems, traveling salesperson and minimum vertex cover, and study the effectiveness of different bet-and-run strategies. In particular, our restart strategies do not take any problem knowledge into account, nor are tailored to the optimization algorithm. Therefore, they can be used off-the-shelf. We observe that state-of-the-art solvers for these problems can benefit significantly from restarts on standard benchmark instances.

3.31 Reducing the size of large instance repositories

Markus Wagner (University of Adelaide, AU)

Over the years, some repositories of test instances have grown significantly. Obviously, there are many more-or-less-biased ways to reduce a given set: based on algorithm performance (e.g. pick the ones where I beat my competition), based on instance features (e.g. largest 10%), and so on. Do we want to represent the distribution of the instances only, or also the density? Long story short: what is the least-biased way to reduce a given repository, and to which extent can we define the faithful (?) subset selection problem. Bonus problem: how to deal with holes in the spaces? Obviously, there is some existing work on very concrete aspects out there... how for up in generality can we go? What would generic algorithmic approaches be?

Among other, this talk gave rise to concepts like instance portfolios, marginal contribution of instances to a portfolio, and to a cycle of algorithm configuration (on instances) and instance generation (for algorithms).

3.32 Accelerating Algorithm Development

Simon Wessing (TU Dortmund, DE)

Main reference S. Wessing, "Towards a Systematic Development Process of Optimization Methods", arXiv:1603.00001v2 [math.OC], 2016.
URL https://arxiv.org/abs/1603.00001v2

Many pitfalls are lurking in algorithm engineering, and it seems that more often than not, they are not related to the solution of the mathematical problem, but to formulating the problem, implementing the algorithm, experimenting with it, and applying it to the real world. These issues typically do not get the attention of scientific research, but may severely bias its outcomes. I give examples where this has happened and try to give recommendations on how to avoid such problems in the development process, with a main focus on the planning of experiments with pre-design experiment guide sheets.

4 Breakout Sessions and Working Groups

4.1 All you ever wanted to know/ask a theoretician and All you ever wanted to know/ask a practitioner

Anne Auger (INRIA Saclay – Orsay, FR) and Carola Doerr (CNRS and University Pierre & Marie Curie – Paris, FR)

The objective of this breakout session was to gather theoreticians and practitioners to (1) better appraise the relevant questions that can or should be addressed by theoreticians for helping practitioners getting insights into the working principles of algorithm selection and configuration and (2) for practitioners to learn where already existing theoretical results could be beneficial in their research activities. The practitioners who were present are mainly working on algorithm configuration and selection while the theoreticians had mostly experience on evolutionary computation methods (including online adaptive methods). Given this difference of background, a substantial amount of time was spend on understanding the problematic on algorithm selection. More precisely, we have mostly discussed how empirical work could influence theory in algorithm configuration and selection. It has been suggested that, for example, in algorithm configuration quite often features that are seen to have a substantial impact on the performance of an algorithm are not very well understood. It could be beneficial for both theoreticians and practitioners to shed light on these effects. To this end, we have discussed potential problems. It is commonly agreed on that analyzing feature-based performance is probably out of reach for NP-hard problems like MAX-SAT and others. On the other hand, even results for much easier problems and algorithms are currently missing.

It is mentioned that not very often statistical models are used as a starting point for theoretical investigations. We discuss why this is the case (typically regarded problems are very difficult to analyze rigorously while the more easily analyzed problems are less interesting for researchers in the more empirical domains) and that theory for simple models could already be quite insightful.

During the discussion Tim Roughgarden's work on "A PAC approach to application-specific algorithm selection" is mentioned as a theory-based study of algorithm selection problems. Furthermore, it is mentioned that in the SAT community the exchange of ideas between empirically and theoretically oriented researchers works quite well. A possible reason for this is the fact that the people agree on the problems that are analyzed and that they agree on common terminology. This is unfortunately not the case in algorithm selection and configuration where even the terminology needs to be agreed upon.

An idea emerging from this breakout session is to try algorithm selection and configuration with components that are "hand-picked" from theoreticians, e.g., by selecting only those which have been analyzed with mathematical rigor and to compare the results that one can achieve with such an approach with those being obtained without any restriction. The gap between such figures could serve as an interesting starting point for further discussions and investigations.

Acknowledgement. Besides the session chairs, the session was attended by Thomas Bäck, Benjamin Doerr, Marcus Gallagher, Wanru Gao, Holger Hoos, Frank Hutter, Lars Kotthoff, Kevin Leyton-Brown, Andres Munoz Acosta, Nysret Muliu, Frank Neumann, Marc Schoenauer and Hao Wang; we are very thankful for their valuable contributions to the discussion.

4.2 Controlled Problem Instance Generation

Marcus Gallagher (The University of Queensland – Brisbane, AU)

In this flex-time session, we discussed current work, open problems and future directions in controlled (aka targeted or strategic) problem instance generation, as a key component in the evaluation of algorithm selection and configuration techniques. A photo of the whiteboard after the session is attached.

Conceptually, for combinatorial. discrete, continuous (or mixed?) black-box optimization problems, we are interested in the performance of algorithm instances (e.g. from an algorithm selection or configuration technique) over some set of problems. The set of all problems may not be interesting (as implied by the No Free Lunch Theorems), since performance (for many definitions) is equal, on average. However many of these problems are uninteresting (e.g. "white noise"): we are really interested in the subspace of problems with some sort of exploitable structure, and/or those with relevance to real-world problems. It is currently unclear to what extent commonly used benchmark problem sets (e.g. BBOB for continuous problems) represent the set of "interesting" problems.

For algorithm evaluation, it is desirable to have good coverage of the set of "interesting" problems when performing experiments. If different algorithms perform well/poorly on different types of problems, good coverage is important to get a clear picture of this. However this raises the question of what is meant by "types of problems". A good part of our discussion was consequently about problem features – since we need ways of measuring the characteristics of problems to identify or measure "problem type". Good features should help us gain a better understand of algorithm performance. It might also be possible to perform better experiments if test problems can be generated which vary smoothly with respect to problem features.

Several different possibilities were discussed regarding controlled problem instance generation:

- Evolving problems using Genetic Programming. Here we have a symbolic representation for the problems – representation is clearly a general, important issue.
- Use of a heuristic search algorithm (e.g. 1+1-EA) to generate problems with some desirable property in feature space.
- Finding real-world representative problems.
- Using surrogate or generative models.
- Blending known functions.

Issues around "feature selection" were also discussed. How many features are needed and how to select them is a issue from Machine Learning. When features are based on sampling the fitness landscape, the sampling technique and size becomes important. The curse of dimensionality suggests that more features require a much larger sample size to support effective estimation.

The session identified many interesting and important issues that would make fruitful research directions.

Acknowledgement. Besides the session chairs, the session was attended by Wanru Gao, Pascal Kerschke, Lars Kotthoff, Andres Munoz Acosta, Samadhi Nethmini Nallaperuma, Mike Preuß, and Heike Trautmann; we are very thankful for their valuable contributions to the discussion.

4.3 Describing / Characterizing Landscapes of multiobjective Optimization Problems

Pascal Kerschke (Universität Münster, DE)

This is a summary of the breakout session on *Describing / Characterizing Landscapes of multiobjective Optimization Problems*, which was held at the *Dagstuhl Seminar 16412 on Automated Algorithm Selection and Configuration* on October 13, 2016.

Initiated by recent research results, which started to characterize multiobjective optimization problems, this breakout session was originally intended to discuss the following four issues:

1. What are characteristics / landmarks / properties of a multiobjective landscape?
2. How can we measure the interaction of the objectives (in addition to simply using indicators)?
3. What could be (cheaply computable) features of a multiobjective landscape?
4. Are there differences across the different domains (continuous, discrete, TSP, etc.)?

However, during the roughly 60-70 minutes of brainstorming, the approx. 10 participants of this session actually re-defined this session. So, we started with some points of the list from above and then the discussion took off...

Measuring the Similarity Between the Objectives

In a first topic, the participants discussed ways to measure the interaction of the objectives (i.e., issue 2 from the original questions). The initial idea was to measure the covariances and/or correlations between the objectives and to somehow estimate the underlying multivariate distribution. In the single-objective scenario, there seem to exist related approaches, called *density of states*. But what would such a similarity information tell us?

Another idea was to have a look at the contour lines of the points in the objective space and somehow use that information to differ between local and global optima.

Detection of Disconnected Pareto Fronts

Based on the idea of analyzing the Pareto fronts, we came to a discussion on whether it is possible to detect disconnected Pareto fronts. One possible solution was to have a look at the objective-wise gradients and see whether they point in opposite directions. Another idea was to apply a clustering approach to the points of the objective space and use the number of found (non-dominated) clusters as representatives of disconnected (global) fronts.

Aggregating the Objectives

In a next approach, Carlos Fonseca introduced us to the basic idea of his PhD-thesis, in which he represented the multiobjective landscape by a single-objective one.

Inspired by the idea of reducing the multiobjective problem to a single-objective one, we came up with the idea of computing the landscape features objective-wise and aggregating them afterwards. But then the question would be how to aggregate that information in a meaningful way and what does it tell us?

Also, is it really useful to spend the same amount of function evaluations for each of the objectives? Maybe some of the objectives are rather easy and thus cheap to compute, whereas others are more complex. Therefore, one could also compute surrogate models for each of the objectives and then use these models to compute numerous (hopefully representative) landscape features of the original landscape.

Approaching the Problem From Different Angles

For some reason, we mainly focussed on the objective space and tried to think of approaches on how to characterize the information that's hidden in there. One idea was to make use of features from the TSP domain. So for instance, one could have a look at the points (= cities) in the objective space and compute features based on their distance matrix, MST or convex hull.

There was also the idea to analyze the path of the (TSP) features across the iterations of the optimization algorithms. That is, we run the optimization algorithm and for each generation, we use the current population to compute the (TSP) features and then analyze how they change over the course of time.

Inspired by the cat-like shapes of the objective space (sketched on the whiteboard), there were also two more ideas coming up:

1. There should be new benchmark functions; problems with specific shapes such as the cat-like shape that was sketched on the board.
2. We could (at least for now) skip the landscape features and see how other (promising) approaches perform. So, one idea was to consider the shape of the sampled objective space as image and then use a deep-learning neural network for performing algorithm

selection on the multiobjective problems. This might not give us direct insights into understanding the characteristics of the problems themselves, but they would at least provide a solid base line for feature-based algorithm selection models. And we could afterwards use these results to find problems for which the algorithms behaved differently and then use that information to develop features which could be more promising for describing these differences.

Acknowledgement. Besides the session chairs, the session was attended by Michael Emmerich, Carlos M. Fonseca, Marcus Gallagher, Carlos Ignacio Hernández Castellanos, Frank Hutter, Manuel López-Ibánez, Andres Munoz Acosta, Heike Trautmann, Markus Wagner, Hao Wang and Simon Wessing; we are very thankful for their valuable contributions to the discussion.

4.4 Portfolio-based Methods for Algorithm Selection

Kevin Leyton-Brown (University of British Columbia – Vancouver, CA)

We had a breakout session on portfolio based methods for algorithm selection. The topics we considered were divided into best practices and hurdles. In the former category, here are questions the group considered:

- Which methods do you prefer and why?
- Have you been involved in successful applications
- What do you consider the role of features?
- Describe your experiences with aslib
- Best approaches for deciding what to put in the portfolio
- Good ideas you think others don't know about
- How to detect incorrect algorithm behavior
- What statistical methods are necessary to ensure validity of results?
- How do things change when selecting parallel solvers?

Here are the questions we considered regarding hurdles:

- What doesn't work?
- Do we still need selection in an increasingly parallel world?
- What interesting findings have you not published?
- How important is fancy machine learning? (And, if it's important, what fancy methods do you like?)
- How is selection connected to PIAC (they are the same; selection is subsumed; . . . ?)

Finally, we discussed the big questions the field should grapple with next. Here is a list of topics identified by the group.

- Parallelism. What if you run everything in parallel?
- Tradeoff between restarts and parallel runs
- How to find many, truly complementary algorithms?
- Is there a sense in which two algorithms can be said to be complementary "always"?
- What can we learn from ML literature on ensemble methods?
- Methods for regularizing portfolios

- Tradeoff in black-box continuous optimization between gathering data for use in features and using the same data in the optimization itself
- PIAC: predicting parameter values from continuous space

4.5 What can we learn from algorithm selection data?

Marius Lindauer (Universität Freiburg, DE) and Lars Kotthoff (University of British Columbia – Vancouver, CA)

Main reference B. Bischl, P. Kerschke, L. Kotthoff, M. Lindauer, Y. Malitsky, A. Frechétte, H. Hoos, F. Hutter, K. Leyton-Brown, K. Tierney, J. Vanschoren, "ASlib: A Benchmark Library for Algorithm Selection", Artificial Intelligence, Vol. 237, pp. 41–58, Elsevier, 2016.
URL http://dx.doi.org/10.1016/j.artint.2016.04.003

Selecting a well-performing algorithm for a given problem instance (e.g., a combinatorial problem or machine learning data set) often substantially improves the performance compared to always using the same algorithm. The automation of this process is called automatic algorithm selection [4] which is often implemented by using machine learning models requiring a lot of training data to get decent predictions. This training data mainly consist of two required matrices, i.e., the performance of each algorithm on each instance, and the instance features for each instance. To reduce the burden on algorithm selection developers to collect these data and to provide standardized data for comparison of algorithm selectors[3], the algorithm selection library ASlib [1] was created.

Besides doing algorithm selection experiments on the data in ASlib, the question of the breakout session was what can we further do with these data to get more insights into the algorithm selection problem to solve, and to get insights why and on which instances an algorithm selector performs well.

A first step in this direction is already done in the exploratory data analysis (EDA) provided on the ASlib website[2]. A user gets insights in the performance distributions of each algorithm, performance correlation of pairs of algorithms and distributions of instance features. To extend this data-driven overview, we discussed further plots to show portfolio contributions of algorithms [2], statistical significance tests, feature importance and cost of computing instance features.

Up to now, papers on algorithm selectors often only report how well they perform on all instances but we lack some detailed analyses which instances they perform well on and where they fail to select a well-performing algorithm. To this end, we discussed that it would be nice to have interactive scatter plots showing the performance of the single best algorithm and an algorithm selector on each instance. Furthermore, a new idea is to train an easy-to-interpret decision tree to classify on which instances the algorithm selector performs well. Such plots could be in principle also automatically generated, if ASlib would allow to upload results from algorithm selection experiments (similar to OpenML [5]).

An further open question is how to automatically pass information extracted by the EDA to an algorithm selector. For example, if the EDA already figured out that some algorithms are dominated and not needed for an algorithm portfolio, or which instance features are

[2] http://www.aslib.net

important, the information could be directly exploited for training an algorithm selector. Right now, this process is still mainly manual.

Currently, ASlib provides 17 algorithm selection benchmarks, all from hard combinatorial problems. Whether these benchmarks can be called real-world benchmarks is unknown. However, a mid-term goal of ASib is include even more diverse benchmarks, for example from the continuous optimization community and from the meta-learning community in machine learning.

In summary, ASlib was well-received and provides a lot of untouched potential to learn more about algorithm selection data and the behavior of algorithm selectors on them.

Acknowledgement. Besides the session chairs, the session was attended by Wanru Gao, Holger Hoos, Nysret Musliu, Samadhi Nethmini Nallaperuma, Marc Schoenauer and Markus Wagner; we are very thankful for their valuable contributions to the discussion.

References

1 B. Bischl, P. Kerschke, L. Kotthoff, M. Lindauer, Y. Malitsky, A. Frechétte, H. Hoos, F. Hutter, K. Leyton-Brown, K. Tierney, and J. Vanschoren. ASlib: A benchmark library for algorithm selection. *Artificial Intelligence*, pages 41–58, 2016.

2 A. Fréchette, L. Kotthoff, T. Michalak, T. Rahwan, H. Hoos, and K. Leyton-Brown. Using the shapley value to analyze algorithm portfolios. In D. Schuurmans and M. Wellman, editors, *Proceedings of the Thirtieth AAAI conference ON Artificial Intelligence*, pages 3397–3403. AAAI Press, 2016.

3 M. Lindauer, H. Hoos, F. Hutter, and T. Schaub. Autofolio: An automatically configured algorithm selector. *Journal of Artificial Intelligence Research*, 53:745–778, August 2015.

4 J. Rice. The algorithm selection problem. *Advances in Computers*, 15:65–118, 1976.

5 J. Vanschoren, J. van Rijn, and B. Bischl. Taking machine learning research online with openml. In *Proceedings of the 4th International Workshop on Big Data, Streams and Heterogeneous Source Mining (BigMine)*, volume 41 of *JMLR Workshop and Conference Proceedings*, pages 1–4. JMLR.org, 2015.

4.6 (ELA features for) multimodal optimization

Mike Preuß (Universität Münster, DE)

Main reference K. Kawaguchi, "Deep Learning without Poor Local Minima", arXiv:1605.07110v3 [stat.ML], 2016.
URL https://arxiv.org/abs/1605.07110v3

This session was intended for brainstorming on possible new Exploratory Landscape Analysis (ELA) features that are especially well suited for extracting knowledge from multimodal optimization problems. However, the discussion centered much more on what the basic properties of multimodal problems are, or even more general, how multimodal optimization itself is defined. As the term is relatively young, there are several opinions on the basic ideas and the need for better definitions was expressed. One example is the optimum definition for ridge functions: is it a set of points, or an areal structure? This even holds true for plateaus, which may be considered as a large set of optima or a single optimum (however, this would be far from a mathematically rigid definition). Detection of a plateau seems to be easy for a human, and more difficult (but probably possible) automatically, even in high dimensions. It is also unclear how common such optimum types are for real world problems, but it is expected that they are of some importance.

Generally, we agreed that we need to rigidly define multimodality, funnel, the whole vocabulary used for this kind of optimization, and that based on that, we need some generally accepted measures. For example, how can we express the multimodality of a problem as compared to another in a value? And how the robustness of peaks?

Another general question that was discussed to some extent was the one for the type of peaks we actually want to detect. Are we satisfied with good local optima? How many? The currently employed benchmarks concentrate on multi-global optimization, but the group considers this rather a special case then of general interest. Additionally, taking into account the recent developments in deep learning, it may be even satisfactory to find good saddle points (good in this context means some that are optima in most dimensions but saddle points in few)? Can we derive benchmark functions with saddle points? The positivity of the Hessian would be a good indicator. We expect interesting results from optimization experiments on such problems as saddle points are difficult to escape. At least, an evolutionary algorithm would probably be slowed down and it could happen that this prematurely triggers termination criteria, making the problem even the more difficult.

The saddle points discussion was based on this work: Deep Learning without Poor Local Minima by Kenji Kawaguchi: https://arxiv.org/abs/1605.07110

Acknowledgement. Besides the session chairs, the session was attended by Carlos M. Fonseca, Marcus Gallagher, Pascal Kerschke, Andres Munoz Acosta, Heike Trautmann and Simon Wessing; we are very thankful for their valuable contributions to the discussion.

4.7 Online and Adaptive Methods

Marc Schoenauer (INRIA Saclay – Orsay, FR)

Preliminary

This document is the main outcome of the breakout session about *Online and Adaptive Methods* that took place during Dagstuhl Seminar *Algorithm Selection and Configuration* – October 10-15.

4.7.1 Monday Flex group meeting

The whole discussion started when Thomas Stützle heavily criticized the long-known diagram, originally proposed by Eiben et al. in their 1999 paper in IEEE TEC [1]. Figure 1 is an adaptation of the initial diagram, taken from [2].

Some mathematical framework was proposed to the discussion by Anne Auger, quickly supported by Carlos Fonseca . . . as this was very close to his own proposal, that he presented on the following Tuesday morning during his short talk about dynamical systems [3]. The discussion then tried to instantiate known instances of parameter setting withing this model, modifying the model itself when necessary – and partly continued during the following days.

At the end of the Dagstuhl seminar, all participants had agreed on the following proposal, that is now offered to the whole community in the hope it can be adopted and improved in order to somehow represent all know instances of parameter setting.

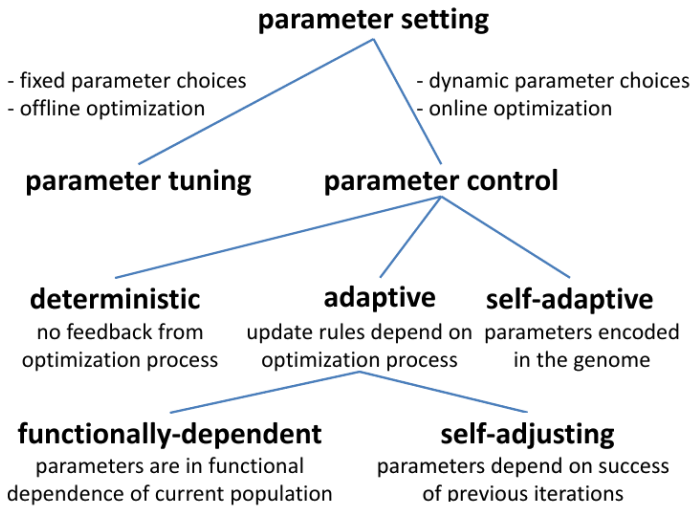

Figure 1 Doerr & Doerr's version [2] of Eiben et al.'s classification diagram [1].

4.7.2 The proposal

- x_t is the population at large (points of the search space). Thus here, $x_t \in \Pi$, space of populations of search points. Note that this includes algorithms with archives, but not algorithms that make use of all population from the very beginning at all steps, as pointed out by Michael).
- σ_t is a set of parameters that are used to update the population, and are themselves updated (or not).
- u_t (and v_t) are uniformly and independently generated random numbers

4.7.2.1 Static parameter setting

$$\begin{cases} x_{t+1} &= F(x_t, \sigma_t, u_t) \\ \sigma_{t+1} &= \sigma_0 \end{cases} \tag{1}$$

Some people (lead by Anne Auger and Benjamin Dörr) suggested to replace u_t with u_{t+1} in this equation. Opponents (Carlos Fonseca, Marc Schoenauer ...) agreed in the end that it is a minor and formal detail.

4.7.2.2 Non-adaptive / feedback-free parameter setting

$$\begin{cases} x_{t+1} &= F(x_t, \sigma_t, u_t) \\ \sigma_{t+1} &= G(\sigma_t, t, v_t) \end{cases} \tag{2}$$

4.7.2.3 Adaptive parameter setting

$$\begin{cases} x_{t+1} &= F(x_t, \sigma_t, u_t) \\ \sigma_{t+1} &= G(x_t, \sigma_t, u_t) \end{cases} \tag{3}$$

with 2 subsets:

4.7.2.4 Functionally dependent

$$\begin{cases} x_{t+1} &= F(x_t, \sigma_t, u_t) \\ \sigma_{t+1} &= G(F(x_t, \sigma_t, u_t)) \end{cases} \tag{4}$$

Note: the second equation could be written as $\sigma_{t+1} = G(x_{t+1})$ but is kept that way to be consistent with the usual form of dynamical system definition.

4.7.2.5 Self-adjusting

This is the part (name and equations) that was most heavily discussed – and no real consensus was reached.

$$\begin{cases} x_{t+1} &= F(x_t, \sigma_t, u_t) \\ \sigma_{t+1} &= G(x_t, u_t) \end{cases} \tag{5}$$

4.7.3 Self-adaptive

The formal definitions above lead to unforeseen difficulties when it came to address self-adaptive properties. Two proposals were made:

- *Hide* the mutation parameters in the definition of the state space, championed by Marc Schoenauer;
- Create some intermediate step to reflect self- property, proposed by Carlos Fonseca.

4.7.3.1 Changing the State Space

The dynamical system formulation is cool – it allows mathematical proofs depending on properties of F and G (see previous work from Anne Auger, and [3] in this document). It should be preserved/extended by relaxing the definition of the first part of the state space where x_t 'lives' (thus meeting the more general formulation that Anne had initially proposed :-)

If you allow the x_t in the above dynamical systems definitions to be something else than a population of search points, this could cover both issues of EDAs and self-adaptive:

- For EDAs, it should be the distribution d_t that evolves. Then function F describes how the distribution is updated (including sampling), and σ the parameters of this sampling/adaptation.
- For self-adaptive algorithms, it should be $\mathcal{S} \otimes \times$, i.e. representing points of the search space to which are attached some parameters (e.g., the mutation parameters in self-adaptive ES, the crossover bit for Spear's GA, etc).

4.7.3.2 Adding Intermediate Stages

For self-adaptation:

$$\begin{cases} \sigma_{t+\frac{1}{2}} &= f_1(\sigma_t, u_t) \\ x_{t+\frac{1}{2}} &= f_2(x_t, \sigma_{t+\frac{1}{2}}, u_t) \\ x_{t+1} &= f_3(x_{t+\frac{1}{2}}) \\ \sigma_{t+1} &= f_4(x_{t+\frac{1}{2}}, \sigma_{t+\frac{1}{2}}) \end{cases} \tag{6}$$

and for self-adjusting:

$$\begin{cases} x_{t+\frac{1}{2}} &= f_1(x_t, \sigma_t, u_t) \\ x_{t+1} &= f_2(x_{t+\frac{1}{2}}) \\ \sigma_{t+\frac{1}{2}} &= f_3(x_{t+\frac{1}{2}}, \sigma_t) \\ \sigma_{t+1} &= f_4(x_t, x_{t+\frac{1}{2}}, \sigma_{t+\frac{1}{2}}) \end{cases} \tag{7}$$

4.7.3.3 Other issues with self-adaptive setting

- Does self-adaptation really work? The nomenclature is unclear, there are differences in opinions about what's adaptive and what's not (see Section 4.7.2).
- The performances of self-adaptation need to be compared to properly-tuned algorithms. In particular
  - Ultimate test: Can self-adaptive method recover the theoretically-best schedule?
  - Two mandatory baselines for all comparisons: random choice, and oracle.

References

1 A.E. Eiben, R. Hinterding, and Z. Michalewicz. Parameter control in evolutionary algorithms. IEEE Transactions on Evolutionary Computation 3(2): 124–141, 1999.

2 B. Doerr and C. Doerr. Optimal Parameter Choices Through Self-Adjustment: Applying the 1/5-th Rule in Discrete Settings. In S. Silva and A.I. Esparcia-Alcázar, editors, *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2015*, pages 1335–1342, 2015.

3 C. Fonseca. Optimisation Algorithm Design: A Control Engineering Perspective. Dagstuhl Seminar on Automated Algorithm Selection and Configuration, 2016.

4.8 Real-world Applications of Meta-Algorithmics

Meinolf Sellmann (IBM TJ Watson Research Center – Yorktown Heights, US)

We discussed industrial and non-profit applications of meta-algorithmics that can be roughly grouped in three different categories:

4.8.1 Automated Model Lifecycle Management

The First and biggest application area we see is the automatic creation, adaptation, and risk management of data science model. This increasingly becomes a need as the availability of more data allows for individualization of predictive and prescriptive models which are deployed and dynamically changing environments. Real-world examples for this technology are, e.g.

1. In Education, the individualization of learning, motivation, content selection, and exercise selection.
2. In Heavy Industries, the management of physical assets.
3. In Medicine, the personalization of healthcare.
4. In Transportation, the provisioning of detailed forecasting models of transportation times in different regions, cities and for different modes of transportation.

In all these domains, the assurance of data science technology will be key to promote wide-spread adoption.

4.8.2 One-of strategic decision support

The second area of application are data science problems that are singular and strategic rather than operational and tactical. Meta-algorithmics can help here to bridge the gap between predictive and prescriptive modeling by automatically providing the ability to phenomenologically study uncertainties in the prediction of predictive models that provide the input for down-stream prescriptive analytics. Moreover, off-line preparation can help learn adaptive control strategies that help guide the decision process as a strategic decision making event unfolds. One example where such a technology could be deployed is in the management of disasters.

4.8.3 Combinatorial Design

Finally, we discussed the problem of assembling given parts to a whole that can be expected to match up will with a given set of requirements. For example, think of a team of experts that needs to be assembled to work on a particular project. Both experts can project are described with certain features, and we need to decide which team configuration will have the best outlook to handle the project well. For such a scenario, portfolio selection techniques are applicable, Moreover, there is a research demand here regarding the design and automatic generation of predictive models that predict team performance.

4.9 Pitfalls and Best Practices for Algorithm Configuration

Marius Lindauer and Frank Hutter (Universität Freiburg, DE)

Automatic algorithm configuration [4] helps developers and users of all types of algorithms to tune their parameters (e.g., options of search heuristics or hyperparameters of machine learning algorithms). This can often substantially improve performance (e.g., running time or prediction error). Applying and comparing automatic algorithm configuration tools (such as *ParamILS* [4], *irace* [8], *SMAC* [3] and *GGA* [1]) is related to empirical benchmarking and can include many subtle pitfalls, even if reliable benchmark libraries such as AClib [6] are used. In the following, we summarize the discussion on this topic in a breakout session at the Dagstuhl seminar "Automated Algorithm Selection and Configuration".

A common mistake in tuning parameters (also in manual tuning and development of algorithms) is to optimize parameters on the same instances that are used later to evaluate their performance. This can lead to overoptimistic performance estimates and over-tuning effects [5]. To avoid this problem, we recommend to first split the available instances into a training and test set, using the training instances for tuning and the test instances to report the performance on. Using an outer cross-validation would estimate the individual performances with lower variance, but it is typically infeasible in algorithm configuration since each single algorithm configuration run is already very expensive. Also, if the computational budget allows for more than one algorithm configuration experiment, we recommend to rather

run experiments on k algorithm configuration benchmarks than a k-fold cross-validation on a single one; this helps to also capture differences across benchmarks.

Related to performance assessment in general, measuring running time can have subtle problems; e.g., it can be influenced by noise induced by other processes running on the same machine. An alternative to measuring running time could be measuring MEMS [7], i.e., the number of memory accesses. MEMS can be measured with a very fine-grained resolution and therefore allow to precisely measure the performance of fast algorithms. To report running time in a publication, an initial study would be necessary to find a mapping from MEMS to running time (which should be a simple linear model).

A wide variety of pitfalls is related to the target algorithm being optimized and the *wrapper* around it (a communication layer between the algorithm configuration procedure and the target algorithm to be optimized). For example, some target algorithms can measure their own running time, but we have experienced that some can also return negative running time. Another example is that some users/configurators blindly optimize for running time without checking that the target algorithm has properly returned; since many algorithms have some bugs, this would often lead to optimizing the running time to crash. Therefore, we recommend to use a uniform and robust wrapper which also handles the running time measurement and resource limitations (e.g., running time or memory limits); our own solution to this is a generic Python wrapper that is easy to instantiate for a given algorithm: https://github.com/mlindauer/GenericWrapper4AC.

Further pitfalls are related to the parameter configuration space defined by value bounds for all parameters of a target algorithm. In order to open up the greatest performance potential, our general recommendation is to include as many meaningful parameters as possible to explore within a fixed computational budget, and to also choose their ranges large enough to prevent most of human bias. However, adding very large bounds (e.g., full 32bit integer ranges) can make it very hard for configurators to find a well-performing parameter configuration. Another pitfall is to add parameters to the configuration space that change the semantics of the algorithm or performance metric; e.g., optimizing the solution quality gap of the mixed-integer programming (MIP) solver *CPLEX* will drastically reduce the running time, but the resulting CPLEX configuration may only return poor solutions of the given MIP problems.

Many other issues were touched on in the session that would go beyond the scope of this short summary. Overall, when working with algorithms, many things can go wrong. As we use automated methods, even more things tend to go wrong. Therefore, it is important to be aware of common pitfalls and best practices to avoid them. In the upcoming book on "Empirical Algorithmics", Holger Hoos [2] lists many useful best practices related to empirical benchmarking. Katharina Eggensperger, Marius Lindauer, and Frank Hutter are currently working on an article that describes best practices and pitfalls in algorithm configuration in more detail.

Acknowledgement. Besides the session chairs, the session was attended by Aymeric Blot, Wanru Gao, Holger Hoos, Laetitia Jourdan, Lars Kotthoff, Manuel López-Ibánez, Nysret Musliu, Günter Rudolph, Marc Schoenauer, Thomas Stützle and Joaquin Vanschoren; we are very thankful for their valuable contributions to the discussion.

References

1 C. Ansótegui, Y. Malitsky, M. Sellmann, and K. Tierney. Model-based genetic algorithms for algorithm configuration. In Q. Yang and M. Wooldridge, editors, *Proceedings of the*

25th International Joint Conference on Artificial Intelligence (IJCAI'15), pages 733–739, 2015.
2 Holger H. Hoos. *Empirical Algorithmics*. Cambridge University Press, 2017. to appear.
3 F. Hutter, H. Hoos, and K. Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In C. Coello, editor, *Proceedings of the Fifth International Conference on Learning and Intelligent Optimization (LION'11)*, volume 6683 of *Lecture Notes in Computer Science*, pages 507–523. Springer-Verlag, 2011.
4 F. Hutter, H. Hoos, K. Leyton-Brown, and T. Stützle. ParamILS: An automatic algorithm configuration framework. *Journal of Artificial Intelligence Research*, 36:267–306, 2009.
5 F. Hutter, H. Hoos, and T. Stützle. Automatic algorithm configuration based on local search. In R. Holte and A. Howe, editors, *Proceedings of the Twenty-second National Conference on Artificial Intelligence (AAAI'07)*, pages 1152–1157. AAAI Press, 2007.
6 F. Hutter, M. López-Ibánez, C. Fawcett, M. Lindauer, H. Hoos, K. Leyton-Brown, and T. Stützle. Aclib: a benchmark library for algorithm configuration. In P. Pardalos and M. Resende, editors, *Proceedings of the Eighth International Conference on Learning and Intelligent Optimization (LION'14)*, Lecture Notes in Computer Science. Springer-Verlag, 2014.
7 Donald E. Knuth. *The Art of Computer Programming, Volume IV*. Addison-Wesley, 2011.
8 M. López-Ibáñez, J. Dubois-Lacoste, T. Stützle, and M. Birattari. The irace package, iterated race for automatic algorithm configuration. Technical report, IRIDIA, Université Libre de Bruxelles, Belgium, 2011.

4.10 Multiobjective Optimisation Algorithm Selection and Configuration

Carlos M. Fonseca (University of Coimbra, PT) and Manuel López-Ibánez (Univ. of Manchester, GB)

This breakout session, which was held on October 11, 2016, aimed at discussing Algorithm Selection and Configuration in Multiobjective Optimisation (MO) contexts. In particular, the following topics were proposed:

1. Selection and configuration of multiobjective optimisation algorithms
2. Algorithm selection and configuration under multiple performance criteria

Automated configuration of multiobjective optimisation algorithms was considered first. It was noted that the outcome of a MO optimisation run is often a set of non-dominated solutions, which makes it difficult to compare such outcomes directly. In the literature, two approaches have been proposed to assess the performance of MO algorithms: quality indicators and the attainment function. Quality indicators map non-dominated point sets onto real values, and make it easy to tune MO algorithms with existing automated configuration tools. However, the choice of quality indicator may influence the tuning process considerably, since different quality indicators may disagree about which of two outcomes is best. It was pointed out that this is particularly noticeable as the number of objectives increases. In contrast, the attainment function approach deals with the distribution of non-dominated point sets directly, and allows the performance of two different algorithms to be compared (to an extent) by means of hypothesis tests, although their power is perceived to be low.

Additionally, it is still not clear how attainment-function based comparisons can be performed when several benchmark problems are used for tuning.

The availability of multiple quality indicators lead to the discussion of the second topic. Unless there is a clear preference for a given indicator, configuring algorithms to perform well with respect to several indicators follows naturally. Questions then arise on how to aggregate information from different indicators, especially when they are conflicting. Using quality indicators at a higher level to aggregate different quality-indicator values was suggested. Another instance of multiple configuration criteria are the runtime and solution-quality views of performance. It was argued that, by including the time at which individual solutions are found in a run as an additional objective, the resulting augmented sets of non-dominated solutions characterise the *anytime* behaviour of the corresponding algorithms. Tuning for anytime performance would then be implemented by applying quality indicators, as before. The issue of how to measure runtime (computing time versus number of function evaluations, for example) was also considered an important issue, with practical effects on configuration results.

Finally, it was felt that the number of established multiobjective benchmark problems is still very limited, despite on-going efforts to address that issue, and that there is insufficient understanding of what multiobjective problem features are relevant, and how their presence may affect algorithm performance. This discussion was continued in another breakout session on the characterisation of the landscapes of multiobjective optimisation problems. Other topics that were identified, but could not be discussed for lack of time, include the tuning of interactive multiobjective optimisation algorithms and the interplay between preference articulation and algorithm selection and configuration.

Acknowledgement. Besides the session chairs, the session was attended by Aymeric Blot, Michael Emmerich, Carlos M. Fonseca, Carlos Ignacio Hernández Castellanos, Laetitia Jourdan, Pascal Kerschke, Marius Lindauer, Manuel López-Ibánez, Samadhi Nethmini Nallaperuma, Thomas Stützle, Heike Trautmann, Markus Wagner and Simon Wessing; we are very thankful for their valuable contributions to the discussion.

Participants

- Anne Auger
INRIA Saclay – Orsay, FR
- Thomas Bäck
Leiden University, NL
- Aymeric Blot
INRIA Lille, FR
- Benjamin Doerr
Ecole Polytechnique – Palaiseau, FR
- Carola Doerr
CNRS and University Pierre & Marie Curie – Paris, FR
- Michael Emmerich
Leiden University, NL
- Carlos M. Fonseca
University of Coimbra, PT
- Tobias Friedrich
Hasso-Plattner-Institut – Potsdam, DE
- Marcus Gallagher
The University of Queensland – Brisbane, AU
- Wanru Gao
University of Adelaide, AU
- Carlos Ignacio Hernández Castellanos
CINVESTAV – Mexico, MX
- Holger H. Hoos
University of British Columbia – Vancouver, CA
- Frank Hutter
Universität Freiburg, DE
- Laetitia Jourdan
INRIA Lille, FR
- Pascal Kerschke
Universität Münster, DE
- Lars Kotthoff
University of British Columbia – Vancouver, CA
- Kevin Leyton-Brown
University of British Columbia – Vancouver, CA
- Marius Lindauer
Universität Freiburg, DE
- Manuel López-Ibánez
Univ. of Manchester, GB
- Andres Munoz Acosta
Monash University – Clayton, AU
- Nysret Musliu
TU Wien, AT
- Samadhi Nethmini Nallaperuma
University of Sheffield, GB
- Frank Neumann
University of Adelaide, AU
- Mike Preuß
Universität Münster, DE
- Günter Rudolph
TU Dortmund, DE
- Horst Samulowitz
IBM TJ Watson Research Center – Yorktown Heights, US
- Marc Schoenauer
INRIA Saclay – Orsay, FR
- Meinolf Sellmann
IBM TJ Watson Research Center – Yorktown Heights, US
- Thomas Stützle
Free University of Brussels, BE
- Heike Trautmann
Universität Münster, DE
- Joaquin Vanschoren
TU Eindhoven, NL
- Markus Wagner
University of Adelaide, AU
- Hao Wang
Leiden University, NL
- Simon Wessing
TU Dortmund, DE