Sampling strategies in automated algorithm configuration

Anastacio, M.I.A.

1

Introduction

When faced with a new problem, a common approach is to transform it into a mathematical representation, which can then be solved using the laws of mathematics. We learn to do so from the early years of primary school and, as the years pass, we gather methods to solve increasingly complex mathematical problems. Among those more complex problems, nondeterministic polynomial time (NP) hard problems are a family commonly considered non-solvable in polynomial time. While creating the correct solution from scratch is typically intractable, verifying if a solution is correct does not require as much time, which leads researchers to develop heuristic methods. Through approximations, trials and errors, those methods are often able to solve NP-hard problems, even large ones, within an acceptable time frame. Not only are heuristic algorithms powerful, they are also configurable: they come with various parameters, corresponding to many switches and knobs one can play with until the fastest path to a solution is found, and they can be tailored to the specificity of the problem instance at hand.

However, to optimise the performance of such algorithms by setting the values of their parameters requires expert knowledge, a lot of time, or often both. Reproducing the exact same process as described in the previous paragraph, researchers took this new optimisation problem, represented it mathematically as the automated algorithm configuration (AAC) problem, and developed heuristic algorithms (referred to as configurators) to configure their heuristic algorithms (referred to as solvers). The configurator will search the space of possible values for the parameters of the solver and evaluate their impact on its performance on the instances of interest, according to a specific performance measure (often the running time of the solver or the problem-specific quality of the solution it could find). Tackling the AAC problem requires searching a large space of configurations and performing time-consuming solver runs on instances. From local search methods to model-based approaches , state-of-the-art configurators are complex search algorithms tailored to mitigate those two difficulties.

In this thesis, we examine the elements of the AAC problem and the decisions made in past works regarding sampling new values or instances. We investigate sampling approaches – in theory, configurator-agnostic – and hypothesise that different approaches would allow to more efficiently search for high-performing configurations by focusing the search and reducing the time required to evaluate the configured solver.

1.1 Background

As mentioned before, one of the most challenging family of problems obtained by representing mathematically real-world problems are the NP-hard problems. Finding algorithms and developing software that can efficiently solve these is an active area of scientific research, due to their high complexity and wide range of applications (see *e.g.* Burch et al., 1994; Matai et al., 2010; Verbert et al., 2017). New solvers for NP-hard problems are regularly developed and compete with each other at yearly competitions, such as the SAT competition (see *e.g.* Heule et al., 2024), the SMT competition (see *e.g.* Weber et al., 2019) and the planning competition (see *e.g.* Taitler et al., 2024). Among these solvers, many can be tailored to a specific application domain through parameters that influence their inner workings and thus their performance. However, manually tuning those parameters is a lengthy and tedious process that relies heavily on expert knowledge. From the early 1990s onwards, researchers began to explore approaches to automatically configure search heuristics (Minton, 1993) and thus solvers based on such heuristics, which later gave rise to the field of AAC and the paradigm of programming by optimisation (Hoos, 2012b). AAC is particularly beneficial when confronted to a large set of similar problems on which the same configuration is expected to perform similarly well. In recent decades, many automated methods to configure – or tune – those solvers have been proposed (see *e.g.* Ansótegui et al., 2009; López-Ibáñez et al., 2016; Pushak and Hoos, 2020). Their great success encouraged researchers to apply them to a range of other algorithms from areas such as material science (see *e.g.* Packwood et al., 2017; Wahab et al., 2020), economics (see *e.g.* Balcan et al., 2018) and process mining(see *e.g.* Ramos-Gutiérrez et al., 2021). When applied to machine learning (see *e.g.* Bergstra and Bengio, 2012; Li et al., 2018), this gave rise to the nowadays prominent research area of automated machine learning (autoML) (Hutter et al., 2019).

1.2 Outline and research questions

Chapter 2 introduces in more detail the problem at hand, explains the diverse approaches that have been used to tackle it in the literature, and describes the configuration scenarios that will be used in the remainder of the thesis.

RQ1 (Chapter 2) How can the configuration scenarios be best described and characterised?

To better understand the challenges of AAC, we examine widely studied configuration scenarios and attempt to characterise them with the goal of facilitating principled comparisons.

Chapter 3 describes the protocol we follow to compare the performance of configurators. It then compares the configurators introduced in the previous chapter and presents insights regarding their strengths and weaknesses.

RQ2 (Chapter 3) How do state-of-the-art configurators compare to each other on a variety of scenarios?

We run various state-of-the-art general-purpose algorithm configurators on the scenarios that we previously identified. We evaluate them in terms of aggregate performance and complementarity across diverse scenarios.

RQ3 (Chapter 3) How do scenario characteristics influence the performance of state-of-the-art configurators?

We relate the performance to the characteristics we defined in Chapter 2 and draw general conclusions about their strengths and weaknesses.

Chapter 4 explores the extent to which the default parameter values are used in current configurators and proposes an approach to make better use of them.

RQ4 (Chapter 4) How and to which extent do current configurators use the default values usually provided by algorithm developers?

Algorithms typically come with a default configuration. We investigate how frequently configurators use this configuration and their impact on performance.

RQ5 (Chapter 4) How can we make better use of known good parameter values – in our case, the default value – to guide the search strategy?

We evaluate a simple strategy to prune the search space around the default value. Based on its success, we propose a sampling strategy that focuses the search around the default values and evaluates its efficiency in a prominent configurator.

Chapter 5 proposes approaches to compare the performance of algorithms against each other while minimising time spent on non-informative problem instances. It is used as a first step towards the comparison of several configurations studied in the following chapter.

RQ6 (Chapter 5) How can we smartly select on which instances to run our evaluation to lower the time spent evaluating bad algorithms?

We evaluate several metrics to select a subset of instances for comparison between two algorithms. Our goal is to make a decision to discard less promising challengers faster.

Chapter 6 evaluates those approaches to compare the performance of several configurations of a single algorithm and integrates them inside a configurator.

RQ7 (Chapter 6) How can we smartly select on which instances to run our evaluation to lower the time spent evaluating bad configurations?

We apply the previously developed metrics to select a subset of instances for comparison between two configurations. To do so, we define two comparison situations that arise in a configurator and evaluate the selection metrics for both.

RQ8 (Chapter 6) How can instance selection boost the configurator performance or speed?

We integrate the most efficient metrics in a configurator to speed up the comparison of configurations and allow more configurations to be compared to the incumbent.

1.3 Publications based on material in this thesis

Parts of this work have given rise to peer-reviewed publications. We give below a short overview of the content of each publication.

Anastacio, M., Luo, C., and Hoos, H. (2019). Exploitation of default parameter values in automated algorithm configuration. In *Workshop Data Science meets Optimisation, DSO, in conjunction with IJCAI*.

In this work, we studied the impact of the given default on the prominent configurators and introduced a naïve approach to focus the search on the default value without needing to make any changes in the configurators. To do so, we reduced the search space around the default value and showed that, for 15 out of 20 scenarios, the prominent configurator SMAC found better configuration on the reduced search space. This work is covered in Chapter 4.

Anastacio, M. and Hoos, H. H. (2020a). Combining sequential model-based algorithm configuration with default-guided probabilistic sampling. In *GECCO 2020: Genetic and Evolutionary Computation Conference 2020, Companion Volume*, pages 301–302. ACM.

This extended abstract laid the foundations of a sampling method integrated inside of the configurator SMAC to follow truncated normal distributions centered around the default values of the parameters. This work is covered in Chapter 4.

Anastacio, M. and Hoos, H. H. (2020b). Model-based algorithm configuration with default-guided probabilistic sampling. In *Proceedings of Parallel Problem Solving from Nature - PPSN XVI, Part I*, volume 12269 of *Lecture Notes in Computer Science*, pages 95–110. Springer.

In this paper, we tested the sampling method presented previously and compared the obtained configurator SMACPS to the two prominent configurators SMAC and irace, reaching better configurations than both of them on more than half of the 16 studied scenarios. This work is covered in Chapter 4.

Matricon, T., Anastacio, M., Fijalkow, N., Simon, L., and Hoos, H. H. (2021). Statistical comparison of algorithm performance through instance selection. In *Proceedings of the International Conference on Principles and Practice of Constraint Programming, CP*, volume 210 of *LIPIcs*, pages 43:1–43:21. Schloss Dagstuhl - Leibniz-Zentrum für Informatik.

In this paper, we sped up the time required to compare the performance of pairs of solvers for NP-hard problems by automatically selecting on which problem instances they should be evaluated and applying a statistical test to decide when enough evidence has been gathered. This work is covered in Chapter 5.

Anastacio, M., Matricon, T., and Hoos, H. H. (2022). Instance selection for configuration performance comparison. In *Meta-knowledge transfer workshop, in conjunction with ECML-PKDD*.

Building on the previous paper, we studied the impact of instance selection methods to compare configurations of a single algorithm instead of comparing algorithms. We showed that similarly, the time required to get enough evidence to decide which is better than the other can be significantly lower than when running them on all problem instances. This work is covered in Chapter 6.

Anastacio, M. (2021). Greybox algorithm configuration. In *Proceedings of the International Joint Conference on Artificial Intelligence, IJCAI*, pages 4875–4876.

This extended abstract was prepared when planning the content of this thesis. While the final content deviates slightly from the one presented then, this extended abstract describes at a high-level the content covered in Chapters 4 to 6.

1.4 Other work published by the author

During the thesis project, the author also contributed to other publications that are only loosely related to this dissertation. Their content is not part of this thesis, but, for completeness, these are outlined in the following.

Fokkinga, D., Latour, A. L. D., Anastacio, M., Nijssen, S., and Hoos, H. (2019). Programming a stochastic constraint optimisation algorithm, by optimisation. In *Workshop Data Science meets Optimisation, DSO, in conjunction with IJCAI*.

Latour, A. L. D., Babaki, B., Fokkinga, D., Anastacio, M., Hoos, H. H., and Nijssen, S. (2020). Stochastic constraint optimisation with applications in network analysis (extended abstract). In *International Workshop on Model Counting, MCW, in conjunction with SAT*.

Latour, A. L., Babaki, B., Fokkinga, D., Anastacio, M., Hoos, H. H., and Nijssen, S. (2022a). Exact stochastic constraint optimisation with applications in network analysis. *Artificial Intelligence*, 304:103650.

Latour, A. L. D., Babaki, B., Fokkinga, D., Anastacio, M., Hoos, H. H., and Nijssen, S. (2022b). Stochastic constraint optimisation with applications in network analysis (extended abstract). In *Workshop on Counting and Sampling 2022, in conjunction with FLoC 2022 and SAT 2022*.

These publications are part of a line of research in which automated algorithm configuration was applied to a solver for stochastic constraint optimisation problems. In the framework of his master project, D. Fokkinga used SMAC to optimise the parameters of the approach developed by A.L.D. Latour (then a PhD student).

Pulatov, D., Anastacio, M., Kotthoff, L., and Hoos, H. H. (2022). Opening the black box: Automated software analysis for algorithm selection. In *International Conference on Automated Machine Learning, AutoML*, volume 188 of *Proceedings of Machine Learning Research*, pages 6/1–18. Proceedings of Machine Learning Research PMLR.

Purucker, L. O., Schneider, L., Anastacio, M., Beel, J., Bischl, B., and Hoos, H. H. (2023). Q(D)O-ES: population-based quality (diversity) optimisation for post hoc ensemble selection in automl. In *International Conference on Automated Machine Learning, AutoML*, volume 224 of *Proceedings of Machine Learning Research*, pages 10/1–34. Proceedings of Machine Learning Research PMLR.

These publications arose from the projects of other PhD students, whom the author assisted in conceptualising the methods, implementing them and analysing the obtained results. The first one explored the usage of algorithm source code features in the context of automated algorithm selection, and the second proposed a quality diversity optimisation ensemble selection method to ensemble machine learning models after their hyperparameters have been optimised.

Rogers, J., Anastacio, M., Bernard, J., Chakhchoukh, M., Faust, R., Kerren, A., Koch, S., Kotthoff, L., Turkay, C., and Wall, E. (2024). Visualization and automation in data science: Exploring the paradox of humans-in-the-loop. In *Workshop on Visualization in Data Science VDS, in conjunction with IEEE Visualization and Visual Analytics Conference VIS*.

This publication resulted from a Dagstuhl seminar in which the author actively participated and calls for bringing back humans in the loop in automated data science. It won a best paper award at the Workshop on Visualization in Data Science.

Kalkreuth, R., de França, F. O., Dierkes, J., Anastacio, M., Jankovic, A., Vasicek, Z., and Hoos, H. (2025). Tinyversegp: Towards a modular cross-domain benchmarking framework for genetic programming. In *GECCO 2025: Genetic and Evolutionary Computation Conference, Companion Volume*. ACM.

This work presents a benchmarking tool for genetic programming. It includes a pipeline implemented by the author for optimising the hyperparameters of genetic programming methods.

Gerlach, B., Anastacio, M., and Hoos, H. H. (2025). On the efficiency of training robust decision trees. In *Poster at the Symposium on AI Verification SAIV, adjunct to the International Conference on Computer-Aided Verification CAV*.

Moeini, E., Vox, C., Anastacio, M., Skaf, W., Barachi, M., and Hoos, H. H. (2026). Neural architecture and hyperparameter selection through meta-learning on time series. In *Proceedings of the AAAI Conference on Artificial Intelligence*. AAAI Press.

These publications were the result of master and bachelor projects of students co-supervised by the author. Gerlach *et al.* studies the robustness of decision trees and ensembles thereof. Moeini *et al.* applied meta-learning to jointly predict the architecture and hyperparameters one should use on a new time series dataset.