Bayesian Hyperparameter Optimization with ALASPO (Extended Abstract)

Dave Pfliegler, Tobias Geibinger, and Thomas Eiter

Institute for Logic and Computation, TU Wien, Austria
dave@pfliegler.at
{thomas.eiter,tobias.geibinger}@tuwien.ac.at

1 Introduction

Answer Set Programming (ASP) [1] is a declarative paradigm rooted in logic programming, and is, unlike procedural languages, not used to encode a problem solving algorithm, but rather the problem itself. ASP in combination with the Large Neighbourhood Search (LNS) [14] metaheuristic, which iteratively destroys and repairs parts of a solution to find an overall better result, specifically the Adaptive Large-Neighbourhood ASP Optimiser *ALASPO* [4], has already proven to be effective in tackling typical optimization problems [4, 5]. These problems are then typically optimized by iteratively finding multiple solutions, each better than its predecessor with an objective value closer to the optimum.

ALASPO uses an ASP solver internally to facilitate both the finding of an initial solution as well as the repair step of the LNS. This solver offers an extensive range of solving and grounding parameters that can greatly influence its performance. In addition, ALASPO offers numerous configuration options that control the behaviour of the LNS process, together resulting in a considerable configuration space. Consequently, any exhaustive exploration of different parameters and combinations thereof, to identify the optimal configuration, is infeasible by hand and calls for using an automated approach, namely Hyperparameter Optimization (HPO) [8, 7]. HPO is the process of identifying the best performing parameters for a given function, by evaluating them on trials, thereby reducing the effort of manual experimentation, improving reproducibility, and potentially improving performance through more thorough exploration.

This extended abstract is based on the Master's thesis "Bayesian Hyperparameter Optimization with ALASPO and Applications" by Dave Pfliegler [13], where the Bayesian Optimization framework for Hyperparameter Optimization *SMAC3* [11] was integrated into ALASPO, to automatically find optimal configurations. The main contributions of the thesis are:

(i) The design and integration of a customizable Bayesian Hyperparameter Optimization approach into the ALASPO framework using SMAC3, to find best performing configurations.
(ii) A customizable clustering of problem instances based on their characteristics, enabling the tailoring of configurations to homogeneous groups of instances, exploiting their similar optimization behaviour, for an overall heterogeneous set of instances.

(iii) An experimental evaluation of the effectiveness of the newly integrated features on a set of four benchmark problems of high industrial relevance.

For space reasons, we will only give a brief overview of the extension of ALASPO in this abstract and refer to the thesis for details. Similarly, the experimental results have been briefly summarized in the appendix, but can be found in full in the publicly available thesis.

2 Hyperparameter Optimization in ALASPO

We integrate the SMAC3 [11] framework for Bayesian Hyperparameter Optimization into ALASPO, to automatically tune ALASPO configurations for a varied set of instances. We want to, among other things, allow a workflow where we have a small set of diverse instances for a problem, and supply them to the tuning process. ALASPO then clusters these instances based on their characteristics, and produces a best configuration for each. Furthermore, the finished tuning process produces an artifact that can be used for future invocations of ALASPO with new instances to automatically select a fitting configuration.

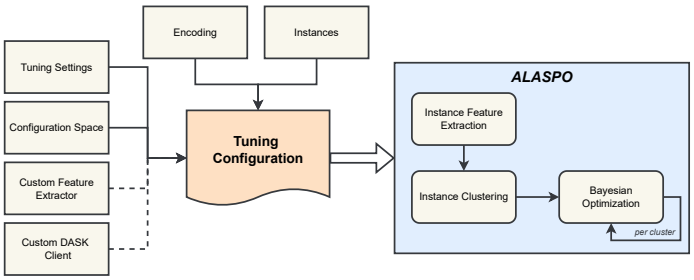

Fig. 1. Overview of the automated tuning in ALASPO.

Figure 1 shows the general tuning approach in ALASPO: The tuning configuration file contains general settings such as the total trial budget, the time limit per run, the configuration space for the optimization, as well as the problem encoding and the instances to be considered. Optionally, a custom feature extractor or a custom DASK configuration can be provided. ALASPO then uses its feature extractor (either the default or a custom one) to obtain a set of features for each instance. This dataset is then used to cluster the provided instances, so that instances with similar characteristics are grouped together. Finally, Bayesian optimization is performed on each instance cluster, resulting in a best found configuration for each. ALASPO will return a `clustering.dill` file after the tuning is finished, which can then be used for future ALASPO invocations, automatically applying a best found configuration based on the given (unseen) instance.

References

1. Brewka, G., Eiter, T., Truszczyński, M.: Answer set programming at a glance. Communications of the ACM **54**(12), 92–103 (Dec 2011). https://doi.org/10.1145/2043174.2043195
2. Chen, J., Li, J., Huang, Y., Garrett, C., Sun, D., Fan, C., Hofmann, A., Mueller, C., Koenig, S., Williams, B.C.: Cooperative task and motion planning for multi-arm assembly systems (2022), https://arxiv.org/abs/2203.02475, https://arxiv.org/abs/2203.02475
3. Dantzig, G., Fulkerson, R., Johnson, S.: Solution of a large-scale traveling-salesman problem. Journal of the Operations Research Society of America **2**(4), 393–410 (Nov 1954). https://doi.org/10.1287/opre.2.4.393
4. Eiter, T., Geibinger, T., Higuera, N., Musliu, N., Oetsch, J., Stepanova, D.: ALASPO: An adaptive large-neighbourhood ASP optimiser. In: Proceedings of the Nineteenth International Conference on Principles of Knowledge Representation and Reasoning (KR 2022). International Joint Conferences on Artificial Intelligence Organization (Jul 2022). https://doi.org/10.24963/kr.2022/58
5. Eiter, T., Geibinger, T., Higuera Ruiz, N., Musliu, N., Oetsch, J., Pfliegler, D., Stepanova, D.: Adaptive large-neighbourhood search for optimisation in answer-set programming. Artificial Intelligence **337**, 104230 (2024). https://doi.org/https://doi.org/10.1016/j.artint.2024.104230, https://www.sciencedirect.com/science/article/pii/S0004370224001668
6. Giustolisi, O., Savic, D.: Optimal design of isolation valve system for water distribution networks. In: Water Distribution Systems Analysis 2008. pp. 1–13. American Society of Civil Engineers (Apr 2009). https://doi.org/10.1061/41024(340)31
7. King, R.D., Feng, C., Sutherland, A.: Statlog: Comparison of classification algorithms on large real-world problems. Applied Artificial Intelligence **9**(3), 289–333 (May 1995). https://doi.org/10.1080/08839519508945477
8. Kohavi, R., John, G.H.: Automatic parameter selection by minimizing estimated error. In: Machine Learning Proceedings 1995, pp. 304–312. Elsevier (1995). https://doi.org/10.1016/b978-1-55860-377-6.50045-1
9. Li, J., Hoang, T.A., Lin, E., Vu, H.L., Koenig, S.: Intersection coordination with priority-based search for autonomous vehicles. Proceedings of the AAAI Conference on Artificial Intelligence **37**(10), 11578–11585 (Jun 2023). https://doi.org/10.1609/aaai.v37i10.26368
10. Li, J., Tinka, A., Kiesel, S., Durham, J.W., Kumar, T.K.S., Koenig, S.: Lifelong multi-agent path finding in large-scale warehouses (2021), https://arxiv.org/abs/2005.07371, https://arxiv.org/abs/2005.07371
11. Lindauer, M., Eggensperger, K., Feurer, M., Biedenkapp, A., Deng, D., Benjamins, C., Ruhkopf, T., Sass, R., Hutter, F.: Smac3: A versatile bayesian optimization package for hyperparameter optimization. Journal of Machine Learning Research **23**(54), 1–9 (2022), http://jmlr.org/papers/v23/21-0888.html
12. Mischek, F., Musliu, N.: The Test Laboratory Scheduling Problem. Technical Report CD-TR 2018/1 (Nov 2018)
13. Pfliegler, D.: Bayesian Hyperparameter Optimization with ALASPO and Applications. Master's thesis, Technische Universität Wien, Wien (2025). https://doi.org/10.34726/hss.2025.124490
14. Shaw, P.: Using constraint programming and local search methods to solve vehicle routing problems. In: Lecture Notes in Computer Science, pp. 417–431. Springer Berlin Heidelberg (1998). https://doi.org/10.1007/3-540-49481-2_30

A Background: ALASPO

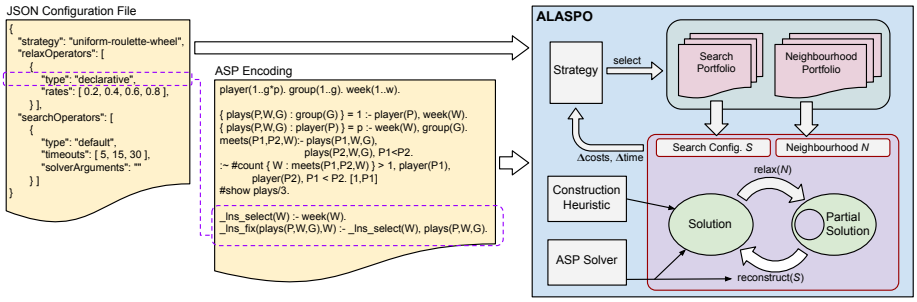

Fig. 2. Adaptive LNS for ASP in the system ALASPO.

ALASPO is a system for ASP optimisation with support for different ASP solvers, search configurations, and neighbourhood definitions. Figure 2 gives an overview of its components and their interaction.

At the heart of ALASPO lies an LNS loop, where an incumbent solution is repeatedly relaxed and reconstructed by an ASP solver to continuously obtain better objective values for the optimisation problem at hand.

An *initial solution* is generated by the ASP solver. Alternatively, it can be obtained by a custom procedure using a construction heuristic, which, however, is problem specific and must be provided via a Python 3 implementation.

In each iteration of the LNS loop, the currently best solution I is relaxed using a *neighbourhood operator* N, which is a procedure to select a subset of the atoms in I. Then, the resulting partial solution is reconstructed using the ASP solver with a constraint to obtain a better objective value than I. This reconstruction depends on a search configuration S which defines solver options and a time limit. If a better solution is found within the time limit, it becomes the new incumbent, otherwise, I remains the best known solution. Which operators are chosen at each iteration, is based on a—potentially self-adaptive—*strategy*.

The optimisation problem is formulated in ASP and stored in one or multiple input files.

A.1 Feature Extraction

To facilitate the clustering of problem instances prior to tuning, each instance must be associated with a set of features produced by a *feature extractor*. The default feature extractor counts the number of occurrences of different facts for each instance, as this is a simple and problem-agnostic way of discerning instances. However, as this approach is highly dependent on how the characteristics of instances are encoded, we allow the implementation of a custom feature extractor that can be tailored to the specific problem at hand to extract more meaningful metrics.

B Experimental Evaluation

For the experimental evaluations, we tested our tuning approach using the same configuration space for all the benchmarking problems, containing all possible ALASPO selection strategies and their respective parameters.

In addition to varying trial budgets, we evaluated the effectiveness of our instance clustering by tuning with different clustering modes. Specifically, we ran the tuning process using a universal default feature extractor, a custom extractor tailored to each problem, and no external clustering, both with and without providing features directly to SMAC3. These modes will be referred to as: *Default Clustering*, *Custom Clustering*, *No Clustering*, and *No Clustering No Features*.

The baseline for the experiments is ALASPO with its default configuration. The thesis provides information on the used encodings and instances, as well as the general experimental setup and methodology.

B.1 Test Laboratory Scheduling Problem (TLSPS)

The Test Laboratory Scheduling Problem (TLSPS) [12] is an extension of the well known *Resource-Constrained Project Scheduling Problem* (RCPSP). It involves multiple projects, each containing a number of jobs that have specific resource requirements, different modes they can be completed in, and a time slot. A solution to this problem is a valid schedule, the quality of which is determined by the sum of multiple soft constraint violations.

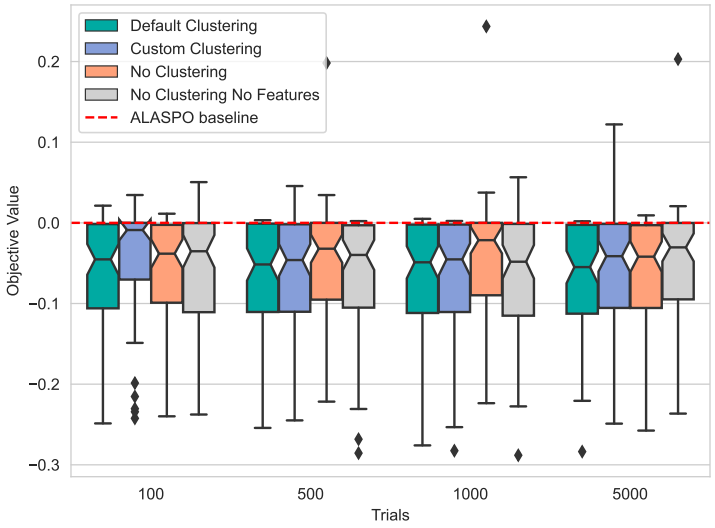

Fig. 3. Tuning results for the TLSPS. Objective values are relative differences to the ALASPO baseline. Detailed results are available in the thesis.

Figure 3 shows the results of the TLSPS tuning. For the TLSPS, tuning the ALASPO strategies resulted in about 7% better objective scores on average, up to 22% for some instances, with virtually strictly equal or better results than the baseline.

B.2 Valves Location Problem (VLP)

Next, we consider the problem of isolation valve placement in water distribution networks [6], which we will refer to as the Valves Location Problem (VLP). It is the practical problem of placing a number of valves in a water distribution network such that when certain pipes need to be isolated from the rest of the system, for maintenance or unplanned interruptions, the disruption to the entire network is minimal.

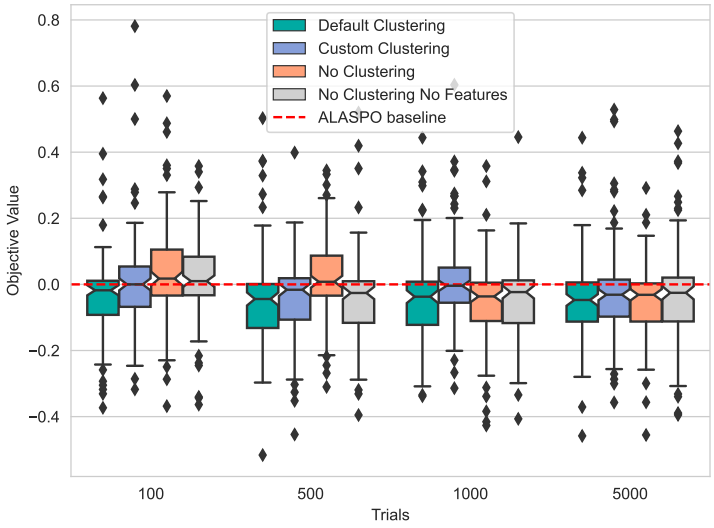

Fig. 4. Tuning results for the Valves Location Problem (VLP). Objective values are relative differences to the ALASPO baseline. Detailed results are available in the thesis.

The results of the tuning for the VLP are shown in Figure 4. Tuning for the VLP resulted in more moderate improvements, with an average of 5% better results, while a quarter of the instances still performed worse than the default configuration.

B.3 Travelling Salesman Problem (TSP)

Tuning for the Travelling Salesman Problem (TSP) [3] – Figure 5 presents the results – resulted in significant performance improvements, with average gains of 28%, as well as strictly better results than the ALASPO baseline for all instances.

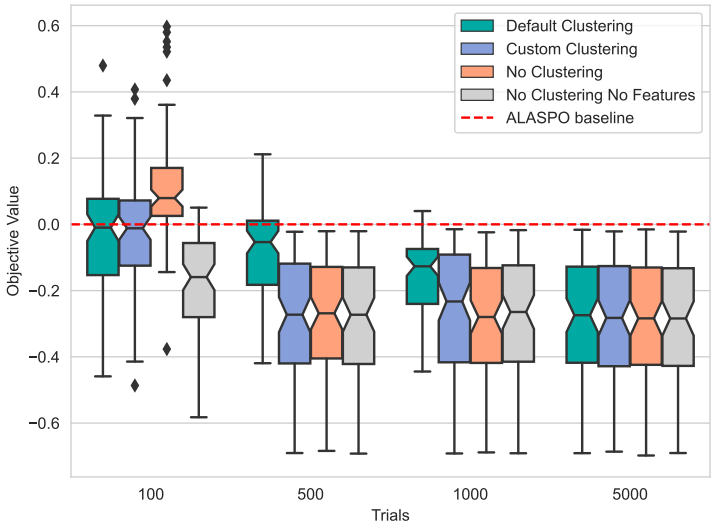

Fig. 5. Tuning results for the TSP. Objective values are relative differences to the ALASPO baseline. Detailed results are available in the thesis.

B.4 Multi-Agent Path Finding (MAPF)

Multi-Agent Path Finding (MAPF) involves finding conflict-free paths for multiple agents sharing the same environment, each having a start position and a goal position they must reach. Naturally, this is a highly relevant problem for industry, with applications in automated warehouse routing [10], robotics [2], and autonomous vehicles [9].

Figure 6 shows the tuning results with the configuration space optimizing the different ALASPO selection strategies and their parameters. Tuning for MAPF could not provide meaningful improvements over the default configuration, suggesting that the ALASPO baseline configuration is already best suited for this problem.

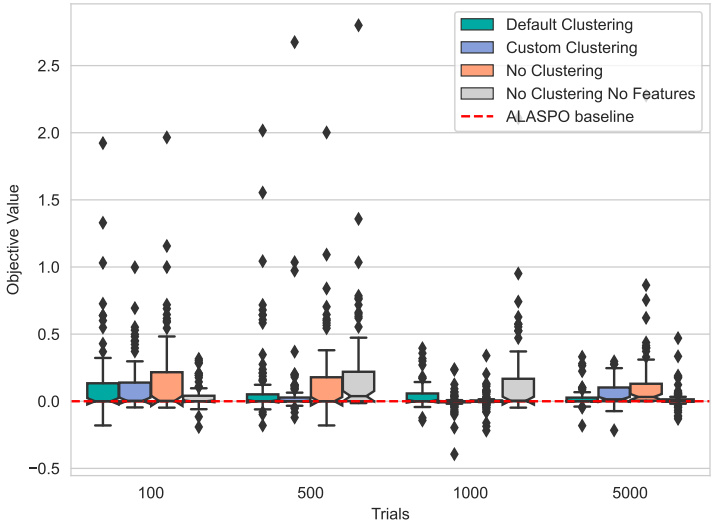

Fig. 6. Tuning results for the MAPF. Detailed results are available in the thesis.