Hyperparameter Optimization of Matheuristics for Hoist Scheduling

Sophie Hildebrandt,a Guido Sanda\*

aPforzheim University of Applied Science, Tiefenbronner Straße 65, 75175 Pforzheim, Germany

guido.sand@hs-pforzheim.de

Abstract

Matheuristics combine mathematical programming with heuristic methods to solve real-world decision problems. The construction of a matheuristic requires to optimize a set of hyperparameters which may be continuous, integer, ordinal or categorial in nature. This paper describes a new approach to construct matheuristics based on Bayesian optimization applied to hoist scheduling problems. First numerical studies compare this approach to the state of the art and quantify their pros and cons.

**Keywords**: hyperparameter optimization, matheuristics, hoist scheduling

* 1. Introduction

Matheuristics are hybrid algorithms that combine mathematical programming with heuristic methods; they can be used to solve real-world decision problems such as hoist scheduling problems. The construction of a matheuristic is a special type of algorithm selection and tuning problem which requires to optimize a set of hyperparameters. While many hyperparameter optimization methods and applications are described in literature, no application of hyperparameter optimization to matheuristics is known to the authors.

This paper describes first insights into hyperparameter optimization of matheuristics using hoist scheduling as an example.

* 1. Hoist scheduling problem

Hoist scheduling problems are flexible job shop scheduling problems with additional hoists; they are proven to be NP-complete (Lei and Wang, 1989). A hoist transports batches of material in carriers between tanks; a plant may have multiple lines of tanks and multiple hoists which run on the same rail and must not collide. A real-world application of hoist scheduling can be found in electroplating plants where ignoble metal is coated with thin layers of noble metal using electrolysis processes (Reimschüssel et al., 2023).

In hoist scheduling, the processing jobs to be scheduled are given by the batches and their recipes; the number and type of tanks and the number and speed of hoists are given as well. The decisions to be taken comprise the sequencing of the batches, the assignment of processing jobs to the tanks and the assignment of transportation jobs to the hoists. A feasible schedule satisfies constraints imposed by the recipes, the plant capacity, the transportation times, and the collision prevention. A typical objective is the minimization of the makespan or the cycle time in case of cyclic schedules.

Hoist scheduling problems can be classified according to the scheme by Manier and Bloch (2003) with respect to the number of lines, the number of transfer systems between the lines, the need to synchronize hoists and transfer systems, the number of hoists, the number and capacity of tanks, the existence of multifunction tanks, the number of (potentially circulating) carriers, the handling system to return empty carriers, the storage for carriers, cleaning of carriers, a loading/unloading station, the number of batches, the number of recipes, the number of reentrant tanks and the number of processing steps.

* 1. State of the art
		1. Construction of Matheuristics

Mathematical programming methods are model based and exact in the sense that they can determine a proven optimal solution of the model in finite computational time. However, if some of the degrees of freedom are subject to integrality constraints, the computational cost often prohibits the practical application to real-world problems. The combination with heuristic methods helps to reduce the computational cost at the price of missing the optimal solution. Research on the combination of mathematical programming methods and heuristics is studied under the headline matheuristics, see e.g. Maniezzo et al. (2009 and 2021). In this context, the term heuristics covers both, population-based metaheuristics like evolutionary algorithms or ant colony optimization, and original heuristics like neighbourhood search and decomposition-based heuristics; the focus of this research is on original heuristics.

The application of matheuristics to hoist scheduling problems was studied in several papers and is still an active research area. Typically, mixed integer programming using general purpose solvers is combined with original heuristics. For instance, Li et al. (2015) combined CPLEX with a zone partition heuristic which limits the range of movement for each of a multitude of hoists. Basán and Méndez (2016) combine GUROBI with a two-phase heuristic combining an incremental construction heuristic with a local improvement heuristic. And Ramin et al. (2023) combined CPLEX with a zone partition and an incremental construction heuristic. The matheuristics are constructed manually or by fully enumerating a predefined grid of potential tuning parameters.

The selection of the heuristic, the tuning of its parameters (e.g. the width of zones in zone partitioning) and the selection of the mathematical programming solver including its parameters are degrees of freedom of the matheuristic construction, but not the only ones: For a given decision problem class like a hoist scheduling problem class multiple mixed-integer programming model formulations exist. For instance, for hoist scheduling problems with one hoist, one line, one recipe, no multi-capacity tanks, and no carrier circulation the models and heuristics described by Basán and Méndez (2016), Chtourou et al. (2013), Feng et al. (2015), Tian et al. (2013), Li et al. (2015), Yan et al. (2018), Zhou and Li (2009) and others are applicable.

* + 1. Hyperparameter optimization

The construction of a matheuristic is a special type of algorithm selection and tuning problem. If its degrees of freedom are considered as hyperparameters of the algorithm, its construction can in principle be automated by means of hyperparameter optimization methods. Various methods for the automated configuration of algorithms are studied in the artificial intelligence domain, see e.g. Schede et al. (2022).

The hyperparameters of an algorithm can be continuous, integer, ordinal or categorical in nature. Examples for hyperparameters in the construction of matheuristics for hoist scheduling problems are the maximum CPU-time for mixed-integer programming solver runs (continuous), the number of tanks assigned to each zone in zone partitioning (integer), the model formulation (ordinal), and the selection of the (potentially not state-of-the art) mixed-integer programming solver (categorical).

Feasible values of hyperparameters are not always independent of each other but may be subject to constraints. For instance, for a given model formulation only particular heuristics are applicable, different heuristics may have different tuning parameters, or the tuning parameters of the heuristic depend on each other. The hyperparameter optimization problem has typically two conflicting objectives: minimize the computational cost and maximize the solution quality of the hoist scheduling problem class given by its objective.

Since no application of hyperparameter optimization to matheuristics is known to the authors, this research strives for first insights into a potential algorithmic setup, useful software tools, and the numerical performance.

* 1. Matheuristics construction using Bayesian optimization

The hyperparameter optimization problem is a black-box optimization problem with expensive-to-evaluate objective functions. Both, the computational cost and the solution quality of the matheuristic are functions of the hyperparameters, which can be evaluated pointwise but provide no prior knowledge like gradients. The high computational cost of each sample is mainly caused by the solution of one or a series of mixed-integer programs.

Hyperparameter optimization methods can be classified as follows: 1. Grid search methods evaluate all combinations of hyperparameter values on a pre-defined multidimensional grid. They are easy to implement but require all grid points to be evaluated. 2. Random search methods distribute the samples randomly in the hyperparameter space. They are also easy to implement but may search in irrelevant areas of the hyperparameter space. 3. Model based optimization methods like Bayesian optimization fit a probabilistic model of the objective functions to the collected function samples and use this model to guide the optimization by trading off exploration versus exploitation of the search space. Model based methods support an efficient sampling but rely on a robust surrogate model which provides useful approximations of the true objective functions, see Bischl et al. (2023).

Due to its efficient sampling strategy, Bayesian optimization is considered the state-of-the-art hyperparameter optimization method for expensive-to-evaluate objective functions like the one at hand (Hutter et al., 2019). It was originally designed for continuous parameters, but integer, ordinal and categorial parameters can be handled as well. A Bayesian optimization algorithm has two main ingredients: 1. An acquisition function balances the exploration versus exploitation of the search space based on the surrogate model and decides which hyperparameter values to evaluate next. Various types of acquisition functions exist, e.g. expected improvement, probability of improvement, and Thompson sampling. 2. The surrogate model is a probabilistic model that defines a distribution over the objective function value in the hyperparameter space between the sampling points. Various types of surrogate models exist, e.g. a Gaussian process, random forest, Bayesian neural networks.

In each iteration step a Bayesian optimization algorithm proposes the next sampling point, determines the corresponding objective function values, and updates the surrogate model. Here, Bayesian optimization is used as an algorithm to automatically construct another algorithm, namely a matheuristic. The main design parameters of a Bayesian optimization algorithm itself are the selection of the acquisition function and of the surrogate model in addition to the setting of their parameters, see Bischl et al. (2023).

* 1. Tool selection and settings

Many commercial and non-commercial implementations of Bayesian optimizations algorithms exist, like GPyOpt (https://sheffieldml.github.io/GPyOpt/), Optuna (https://optuna.org/), scikit-optimize (https://scikit-optimize.github.io/stable/), SMAC3 (https://github.com/automl/SMAC3), Spearmint (https://github.com/JasperSnoek /spearmint) and others. They differ in properties like the programming language (e.g. Python, C++, Julia, R), the spectrum of supported acquisition functions and surrogate models, the ability to handle hyperparameter constraints, extensions (like multi-objective optimization), and the quality of the documentation. For the numerical studies in this paper the non-commercial package SMAC3 was selected, since it supports multiple acquisition functions, surrogate models, all types of parameters (continuous, integer, ordinal, categorial) as well as the handling of hyperparameter constraints.

The entire toolchain was implemented in Python: SMAC3 is a Python-package; it calls the matheuristic (comprising the original heuristics and the mixed-integer programming model of the hoist scheduling problem) which is implemented in Pyomo (www.pyomo.org); the mixed-integer programming problems are solved by CPLEX.

For the SMAC3 design parameters the “black box facade” with expected improvement as acquisition function and Gaussian process as surrogate model was chosen. The standard settings were modified such that multiple evaluations of hyperparameter values are prevented and the initial hyperparameter values are reproducible.

* 1. Numerical studies
		1. Hoist scheduling problem classes and matheuristic

Three small hoist scheduling problem instances from three different classes are studied. They are based on the mixed-integer linear programming model described in detail by Basán and Méndez (2016); the model is limited to a single hoist operating on a single line, tanks with a capacity of one carrier and non-circulating carriers. The objective is to minimize the makespan $MK$ for a given set of jobs $i$, subject to the constraints outlined in section 2 except for the collision prevention. All instances comprise six jobs $i=\{1,…, 6\}$. Instance 1 (taken from Aguirre et al., 2013), comprises seven tanks $j=\{0,…, 7\}$ with one reentrant tank and three recipes $Seq\_{(i)}\in \{1, 2, 3\}$ with a maximum of eight processing steps $s=\{1, …, 8\}$, instance 2 comprises seven tanks $j=\{0,…, 7\}$ with no reentrant tank and three recipes $Seq\_{(i)}\in \left\{1, 2, 3\right\}$ with a maximum of six processing steps $s=\{1, …, 6\}$, and instance 3 comprises five tanks $j=\{0,… , 5\}$ with one reentrant tank and one recipe $Seq\_{(i)}\in \left\{1\right\}$ with $s=\{1, …, 6\}$ processing steps.

The heuristic (from the same paper by Basán and Méndez, 2016) combines a constructive with an improvement step. In the constructive step the first NSJ (number of selected jobs) from a list of jobs are scheduled, the first job is fixed, then the next NSJ jobs are scheduled, and so on. In the improvement step the first NRJ (number of released jobs) of the schedule from the construction step are rescheduled. If no better solution is found, jobs two to NRJ+1 are rescheduled, and so on. If a better solution is found, the improvement step starts from job one. With NSJ=1 and NRJ=1, the constructive and the improvement step consist of at least 6 iterations each. Instead of solving the monolithic model for instance 1, 2 and 3 with 11916/6449, 6469/3607 and 8583/4021 constraints/variables, the model is decomposed into smaller problems. During the constructive step the number of constraints/variables range from 451/223, 191/117 and 248/131 in the first iteration to 11916/5810, 6469/3248 and 8583/3662 in the last iteration.

* + 1. Grid search versus Bayesian optimization

In the first part of the numerical studies, NSJ and NRJ are the (integer-valued) hyperparameters considered. A grid search method is studied as reference.

The setup of the grid search is as follows: NSJ and NRJ range from one to six resulting in grid covering 36 points; the computational time for the evaluation of one point is limited to 3,600 CPU-s; and the two objectives – makespan of the hoist schedule and total CPU-time – are combined into one weighted sum using the (a posteriori calculated) inverse mean values over the 36 points as weighting factors.

The setup of the Bayesian optimization is as follows: The limitation of the computational time and the objective function are the same as above. The algorithm terminates after 16 retries of finding new hyperparameter values, which is the standard setting in SMAC3.

The computational results are summarized in Table 1, sub-tables 1.1, 2.1 and 3.1 for instances 1, 2 and 3, respectively. The grid search certainly finds the global optimum while the best solution found by the Bayesian optimization is not necessarily the global optimum. The Bayesian optimization evaluates only 56% (instance 1), 64% (instance 2) and 58% (instance 3) of the points; it finds the global optimum for instances 2 and 3 and a solution with an optimality gap of 10% for instance 1.

For instances 1 and 3 the best value for NRJ is 6. In these cases, the full monolithic scheduling model is solved in step two of the heuristic, indicating that it would have been better not to apply the heuristic.

* + 1. Extended hyperparameter space

In the second part of the numerical studies, the dimensionality of the hyperparameter space is extended by the number of fixed jobs (NFJ) in the construction step. While NFJ was one in the previous section, all feasible values are considered now. NFJ must not be larger than NSJ such that the constraint NFJ ≤ NSJ applies; consequently, the number of points in the grid search increases to 126.

Due to the results for instance 1 and 3 in the first part (NRJ=6), the studies in the second part are limited to instance 2; the results are summarized in Table 1, sub-table 2.2. Each figure is the best objective value over all NFJ-values. The Bayesian optimization finds the global optimum with only 25% of the points evaluated by the grid search.

Table 1: Objective values of the studied problem instances (points evaluated by Bayesian optimization in bold, optimal value shaded, best value found by Bayesian optimization framed)

|  |  |
| --- | --- |
| 1.1 | NRJ |
| 1 | 2 | 3 | 4 | 5 | 6 |
| NSJ | 1 | 1.56 | **2.07** | 2.22 | **2.08** | 2.08 | 1.46 |
| 2 | **1.68** | 2.9 | **2.34** | 2.21 | 2.20 | 1.58 |
| 3 | **1.78** | **1.84** | **2.04** | 2.62 | 2.28 | 1.66 |
| 4 | **1.74** | **1.81** | 2.00 | **2.58** | **2.25** | 1.62 |
| 5 | **2.15** | **2.06** | **2.28** | 2.14 | 2.03 | **1.72** |
| 6 | **2.05** | **1.95** | 2.18 | **2.03** | **1.92** | **1.61** |

|  |  |
| --- | --- |
| 2.1 | NRJ |
| 1 | 2 | 3 | 4 | 5 | 6 |
| NSJ | 1 | **1.18** | **1.66** | **1.70** | **2.05** | **1.98** | **1.51** |
| 2 | 1.31 | **1.48** | 1.81 | 2.30 | 2.07 | **1.60** |
| 3 | **1.29** | **1.64** | 2.12 | **2.50** | **2.12** | **1.65** |
| 4 | 1.73 | **2.07** | 2.55 | **2.77** | **2.56** | 2.08 |
| 5 | **1.61** | 1.96 | **2.44** | 2.77 | 2.44 | 1.97 |
| 6 | **1.60** | **1.94** | 2.42 | **2.77** | **2.43** | **1.95** |

|  |  |
| --- | --- |
| 3.1 | NRJ |
| 1 | 2 | 3 | 4 | 5 | 6 |
| NSJ | 1 | **1.60** | **1.76** | **2.20** | **2.02** | **1.91** | **1.51** |
| 2 | **1.53** | 1.71 | 2.20 | 2.01 | 1.90 | **1.50** |
| 3 | **1.71** | **1.87** | 2.32 | **2.14** | **2.02** | 1.63 |
| 4 | 1.83 | **1.99** | 2.44 | **2.26** | **2.14** | **1.74** |
| 5 | 1.98 | 2.15 | **2.60** | 2.41 | 2.30 | 1.90 |
| 6 | **1.89** | **2.05** | 2.49 | **2.31** | 2.20 | **1.80** |

|  |  |
| --- | --- |
| 2.2 | NRJ |
| 1 | 2 | 3 | 4 | 5 | 6 |
| NSJ | 1 | 1.18 | 1.66 | 1.70 | 2.06 | 1.98 | 1.51 |
| 2 | 1.29 | 1.48 | 1.81 | 2.30 | 2.07 | 1.59 |
| 3 | 1.26 | 1.60 | 2.02 | 2.46 | 2.09 | 1.62 |
| 4 | 1.29 | 2.05 | 1.93 | 2.38 | 2.09 | 1.62 |
| 5 | 1.14 | 1.49 | 2.02 | 2.36 | 1.98 | 1.51 |
| 6 | 1.60 | 1.94 | 2.42 | 2.77 | 2.43 | 1.95 |

* 1. Conclusions

In this paper, a new method for the automated construction of matheuristics was described and validated for the application to hoist scheduling problems. First numerical studies indicate that the new approach based on Bayesian optimization is preferrable to grid search in case of hyperparameter spaces with higher dimensions.

Future work will elaborate on limitations of the new method including broader numerical studies (e.g. more problem classes), variation of the design parameters (e.g. other acquisition functions and surrogate models), and other types of hyperparameters (e.g. selection of heuristics). Special attention will be paid to multi-fidelity Bayesian optimization. One idea is to steer the precision of the objective function evaluation by varying the termination criterion of the mixed-integer programming solver.

References

A. Aguirre, C. Méndez, A. García-Sánchez, M. Ortega-Mier, 2013, Applying MILP-based algorithms to automated job-shop scheduling problems in aircraft-part manufacturing, “do SimposioArgentino de Informatica Industrial

N. Basán, C. Méndez, 2016, Hybrid MILP/Simulation/Heuristic Algorithms to Complex Hoist Scheduling Problems, CACE, 38, 1929-1934

B. Bischl, M. Binder, M. Lang, T. Pielok, J. Richter, S. Coors, J. Thomas, T. Ullmann, M. Becker, A. Boulesteix, D.Deng, M. Lindauer, 2022, WIREs Data Mining and Knowledge Discovery, 13, 2

S. Chtourou, M. Manier, T. Loukil, 2013, A hybrid algorithm for the cyclic hoist scheduling problem with two transportation resources, Computers & Industrial Engineering, 65, 426-437

J. Feng, A. Che, C. Chu, 2015, Dynamic hoist scheduling problem with multi-capacity reentrant machines: A mixed integer programming approach, Computers & Industrial Engineering, 87, 611-620

F. Hutter, L. Kotthoff, J. Vanschoren (Ed.s), 2019, Automated Machine Learning – Methods, Systems, Challenges, Springer

L. Lei, T. Wang, 1989, A proof: the cyclic hoist scheduling problem is NP-complete. Graduate School of Management, Rutgers University, Working Paper, 89-0016

X. Li, F. Chan, S. Chung, 2015, Optimal multi-degree cyclic scheduling of multiple robots without overlapping in robotic flowshops with parallel machines, Journal of Manufacturing Systems, 36, 62-57

M. Manier, C. Bloch, 2003, A Classification for Hoist Scheduling Problems, International Journal of Flexible Manufacturing Systems, 15, 37-55

V. Maniezzo, T. Stützle, S. Voß, 2009, Matheuristics – Hybridizing metaheuristics and mathematical programming, Springer

V. Maniezzo, M. Boschetti, T. Stützle, 2021, Matheuristics – Algorithms and Implementations, Springer

D. Ramin, D. Fraizzoli, A. Ballarino, A. Brusaferri, Dynamic hoist scheduling for multi-recipe and multi-stage production lines: A logical framework, 2023, Computers & Industrial Engineering, 182,

S. Reimschüssel, U. Fuchs, G. Sand, 2023, Electroplating scheduling: Closing a research gap from an automation vendor’s perspective, CACE, 52, 125-130

E. Schede, J. Brandt, A. Tornede, M. Wever, V. Bengs, E. Hüllermeier, K. Tierney, 2022, A Survey of Methods for Automated Algorithm Configuration, Journal of Artificial Intelligence Research, 75, 425-487

N. Tian, A. Che, J. Feng, 2013, Real-time hoist scheduling for multistage material handling process under uncertainties, AIChE, 59, 1046-1048

P. Yan, S. Liu, T. Sun, K. Ma, 2018, A dynamic scheduling approach for optimizing the material handling operations in a robotic cell, Computers & Operations Research, 99, 166–177

Z. Zhou, L. Li, 2009, A solution for cyclic scheduling of multi-hoists without overlapping, Annals of Operations Research, 168, 5-21