Article

Integrated Simulation-based Optimization of Operational Decisions at Container Terminals

Marvin Kastner [1,†], Nicole Nellen [2,†], Anne Schwientek [3,†], and Carlos Jahn [4]

[1] Hamburg University of Technology, Institute of Maritime Logistics; marvin.kastner@tuhh.de
[2] Hamburg University of Technology, Institute of Maritime Logistics; nicole.nellen@tuhh.de
[3] Hamburg University of Technology, Institute of Maritime Logistics; a.schwientek@tuhh.de
[4] Hamburg University of Technology, Institute of Maritime Logistics; carlos.jahn@tuhh.de
* Correspondence: marvin.kastner@tuhh.de; Tel.: +49 40 42878 4793
† These authors contributed equally to this work.

Abstract: At container terminals, many cargo handling processes are interconnected and take place in parallel. Within short time windows, many operational decisions need to be taken considering both time and equipment efficiency. During operation, many sources for disturbance exist. These are the reason why perfectly coordinated processes are possibly unraveled. An approach that considers disturbance factors while optimizing a given objective is simulation-based optimization. This study analyses simulation-based optimization as a procedure to simultaneously scale the number of utilized equipment and to adjust the choice and tuning of operational policies. The four meta-heuristics Tree-structured Parzen Estimator, Bayesian Optimization, Simulated Annealing, and Random Search guide the simulation-based optimization process. The results show that simulation-based optimization is suitable to identify the amount of required equipment and well-performing policies. Thereby, there is no clear ranking which of the meta-heuristics finds the best approximation of the optimum. The approximated optima suggest that pooling terminal trucks as well as a yard block assignment close to the quay crane is preferable. With an increasing number of quay cranes, the number of optimal terminal trucks for each quay crane decreases as well as the range of truck utilization within one experiment.

Keywords: container terminal; simulation; simulation-based optimisation; meta-heuristic; horizontal transportation;

1. Introduction

Seaports are the interface between various transport modes in the maritime supply chain. Compared to 1997, the volume of global maritime containerized trade tripled to 152 million TEU (Twenty-foot Equivalent Unit, size of a standard container) in 2019 [1]. At the same time, ship sizes have also tripled in the past twenty years from 8,000 TEU capacity to around 24,000 TEU. This implies that — in addition to adjustments to the port's infrastructure and superstructure — container terminals have to substantially increase their efficiency in ship handling in order to keep unproductive berthing times as short as possible while the container volumes to handle during one ship call increase. Thus, the challenge for terminals is to handle a large number of containers within a very short period of time. Terminals can meet this challenge by creating the technical prerequisites (i.e. using more and higher-performance equipment) and by optimizing operational processes. While the use of more equipment entails correspondingly more investment and higher running costs, the intelligent control of operational processes without additional costs leads to a more efficient cargo handling. Therefore, it is reasonable to use the minimum necessary equipment and coordinate operational processes.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 December 2020 doi:10.20944/preprints202012.0194.v1

Container handling requires a large number of process steps in the terminal. When a ship is berthed, quay cranes (QCs) unload the containers and set them down on waiting terminal trucks (TTs). These TTs transport the containers to the storage area. There, rubber-tired gantry cranes (RTGs) lift the containers into the respective yard block (YB) for short-term storage until the container is picked up. The loading process of a ship is consequently running in the other direction. The processes are coupled as TTs are passive equipment that is not able to lift the containers itself. This means that waiting times and utilization of the respective equipment must be weighed against each other. In order to be able to carry out the ship handling as quickly as possible and to reduce the waiting times of the QCs, longer waiting times and a lower utilization of the TTs are often accepted. The better this conflict of objectives is balanced, the more efficiently the terminal can work.

There are several decision problems in the design and operation of container terminals, which strongly influence the efficiency of container handling. Figure 1 shows an overview of typical decision problems at container terminals.

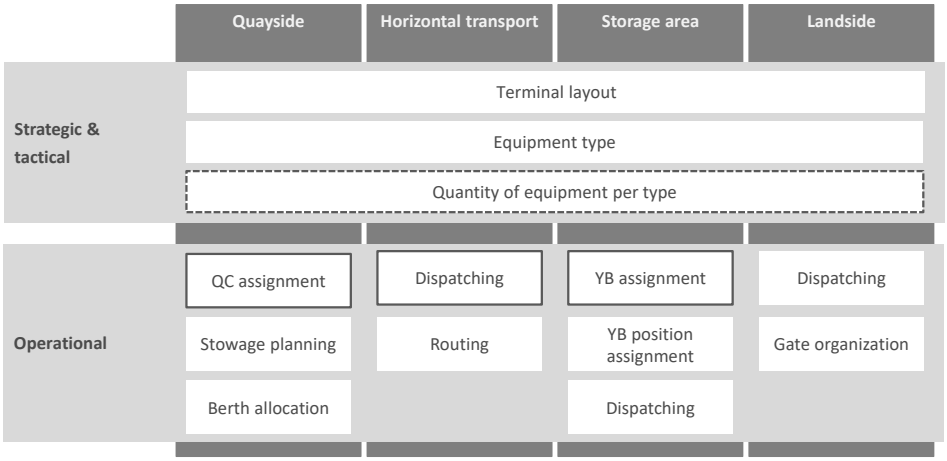


Figure 1. Decision problems at a container terminal

Decisions on how to design the layout of the terminal (e.g. location and size of the YBs) or which equipment should be used and how much of it to procure have a rather long-term influence (refer to [2] for a recent overview). In a short-term perspective, there are on the quay side the berth allocation problem, the stowage planning, and the QC assignment and scheduling problem (refer to [3] for an overview). In horizontal transport, decision problems are dispatching (assigning vehicles and transport orders) and routing. In the storage area, there are the decision problems of dispatching (assigning RTGs and storage orders) as well as the YB assignment and YB position assignment. On the land side there are also the questions of order assignment and gate control. Kizilay and Eliiyi [4] provide a recent overview on container terminal decision problems.

These decision problems influence each other [5]. For example, the berth allocation directly influences the YB assignment and vice versa. The distances between the ship at the berth and the assigned YBs should be as short as possible and at the same time sufficient YBs should be assigned to a berth or QC. RTGs in the yard typically can move 15 containers within one hour while QCs have a productivity of around 30 moves/h. Thus, at least two YBs (with each one RTG) have to be connected to one QC. Another example is the relationship between gate organization and dispatching in the yard: if the number of truck arrivals is regulated by a truck appointment system [6], this also influences the number of handling orders for the RTGs and thus the dispatching. These are just two examples of the numerous interactions between the decision problems. Therefore, there is a risk that the overall solution will deteriorate if only one decision problem is optimized. One solution to this challenge is an integrated optimization, e.g. that the horizontal transport is optimized together with the RTGs and the QCs. At first glance, this is a sensible approach. However, this results in a very complex problem, for

 doi:10.20944/preprints202012.0194.v1

which it is difficult to find a solution and, especially under the real-time requirements of a container terminal, is hardly possible in terms of computing power.

1.1. Previous Methods to Ensure Efficient Operations

There are two main approaches to address operational decision problems of container terminals. The first *dynamic* approach typically uses priority rules, which are analyzed with the help of simulation models. Simulation models on container terminals are reviewed in [7] and [8]. In the second *static* approach, a mathematical problem is formulated which is then either solved optimally for small instances or examined with the help of meta-heuristics.

While the dynamic approach fits better to the volatile processes at container terminals, simulation studies that aim to investigate several decision problems quickly become very complex [9]. The static approach also quickly reaches its limits. Zhen *et al.* [10] show that the integrated QC and TT scheduling problem is NP-hard. This means that the computing time required to solve the problem optimally is too long to be used in production. Especially in order to investigate integrated decision problems that influence each other, the approach of combining simulation and optimization has been developed in recent years to take advantage of both approaches. Kastner *et al.* [11] provide a literature overview on simulation-based optimization at container terminals. They focus on the covered problems, chosen meta-heuristics, and the shapes of the parameter configuration space in the respective publications. Zhou *et al.* [12] present similarly a summary of publications on the integration of simulation and optimization for maritime logistics. They classify five modes of integration depending on the interaction of both techniques. He *et al.* [13] address the integrated QC, TT and RTG scheduling. They develop a mixed integer programming model and propose a simulation-based optimization method. Thereby, their optimization algorithm integrates a genetic and a particle swarm optimization algorithm. Cao *et al.* [14] aim to schedule RTGs and TTs simultaneously in order to decrease the ship turnaround time. They introduce a multi-layers genetic algorithm to solve the scheduling problem and design an algorithm accelerating strategy. Castilla-Rodríguez *et al.* [15] focus on the QC scheduling problem. They integrate artificial intelligence techniques and simulation combining an evolutionary algorithm with a simulation model to embed uncertainty. Legato *et al.* [16] investigate the problem to assign ship bays and QCs as well as sequence discharge/loading operations. Their simulation-based optimization model uses a simulated annealing algorithm for the schedule and a simulation framework for performance estimation. Kizilay *et al.* [17] study the integrated problem of QC assignment and scheduling, YB assignment, and TT dispatching. They propose a mixed integer programming and a constraint programming model and show that the constraint programming model performs much better in terms of calculating time. Sislioglu *et al.* [18] combine discrete event simulation, data envelopment analysis and cost-efficiency analysis to investigate different investment alternatives based on the number of QCs, total length of a quay, TTs and RTGs. The apply their model to 16 different scenarios but do not modify the operating policies. Kastner *et al.* [19] propose to apply the Tree-structured Parzen Estimation (TPE) approach to scale the amount of utilized equipment in a simulation model. With the help of simulation-based optimization, only a subset of the experiments are executed. At the same time, a fine search grid (all equipment is scaled in step sizes of 1) allows a very good approximation for the optimum.

The presented study is an extension of [19]. The reviewed literature is extended and updated. Previously, only the amount of QCs and TTs has been scaled. In this study, in addition the number of YBs is alternated and the coordination of the equipment is varied. That required several extensions at the simulation model. In the new study, the number of QCs is considered fixed during an optimization run. Furthermore, a caching mechanism has been implemented to speed up the optimization study. As a new meta-heuristic, Bayesian Optimization (BO) has been introduced.

The application of dispatching strategies and different policies sets this publication apart from most of the previously mentioned publications. These often directly search for (near-)optimal sequences of container handling tasks. For larger container terminals, Terminal Operating Systems integrate the

 doi:10.20944/preprints202012.0194.v1

computed schedules of different equipment [20]. Especially smaller container terminals tend to use less complex IT solutions that lack automated scheduling methods. They rather rely more on operational rules of thumb, such as dispatching strategies [21]. This study presents a solution method that is applicable for these smaller container terminals. The simulation results describe the (near-)optimal combination of many decisions, such as the number of equipment, the dispatching strategy, and other policies for a given situation. In this study, optimization plays a very different role compared to the above-mentioned scheduling methods. Now several parameters, some of them categorical, some discrete, and some continuous, are adjusted in parallel. It is known from similar prior studies (e.g. [9]) that the different parameters also affect each other. In this study, therefore a multivariate optimization problem is solved.

1.2. Previous Approaches to Optimize Simulation Models

The integration of several operational problems leads to many decisions that are made in parallel. In the presented study, concurrently the number of utilized resources is scaled while different storage policies and equipment control policies are taken into account that themselves sometimes allow policy tuning.

For simulation, often a full factorial design is used where each parameter combination is tried out by running a corresponding simulation experiment [22]. The full factorial design is also called grid search in the machine learning community. A study that covers all parameter variations is only feasible for finite sets such as categorical values or finite sets of numerical values. Theoretically continuous parameters (e.g. real numbers) need to be restricted to a finite set of values. The search grid for such a study needs to be sufficiently fine (i.e. for natural numbers little omissions within a given range, for real numbers using small step sizes) so that the optimum is approximated well. No matter whether it is a simulation study or the parameter tuning part in machine learning, for high-dimensional parameter configuration spaces the combination of all concurrently varied parameters at some point turns out to be too large for exhaustive examination. Even if a grid is exhaustively examined, for real numbers only an approximation for the optimal input parameter might have been identified.

When the parameter configuration space could not be covered, e.g. by time limitations, a subset of possible parameter combinations must be selected. In case of optimization, understanding interactions between different parameters might be a helpful step for finding the optimum. Still, they are not the aim of such a study. There are several approaches to reduce the computational time of a simulation study to a feasible time frame. If the simulation model is sufficiently complex, there is no shortcut to the experiment results — every reduction of the computational time comes with the risk of omitting the true optimum.

One option is to use a multi-fidelity approach [23]. With a low-fidelity simulation model each parameter combination of consideration is evaluated. These results are used to identify the promising parameter configurations that are further examined with a high-fidelity simulation model. From that subset, the best solution can be determined. This approach requires the simulator to create two simulation models of different fidelity. The low-fidelity simulation model requires special skills during creation: all important aspects need to be covered in the model because otherwise a promising parameter configuration for the high-fidelity simulation experiment could be omitted. At the same time the processes must be sufficiently simplified to improve the required time for running the experiments.

Alternatively, a computing budget can be maintained – the total amount of computing resources that are available to approximate the optimal solution [24]. The initial experiments are randomly chosen from the parameter configuration space [25]. If during evaluation a given parameter configuration is identified as better-performing, more computing resources are invested to get a better picture of the corresponding objective function value. The longer the total observed time range for a given simulation model, the more the (typically noisier) sample statistics approximate the population parameters. Al-Salem *et al.* [24] state that this approach does not necessarily create optimal solutions but just solutions that are close to the optimum with a very high probability. Furthermore, in industry

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 December 2020 doi:10.20944/preprints202012.0194.v1

such sufficiently well-designed solutions often satisfy the requirements [24]. Even if optimal input parameters are calculated for a given simulation model, the difference in performance might be of little practical relevance.

Another option is to embed the simulation into an outer loop of optimization. The simulation model itself is regarded as a black-box function[1] and the optimization algorithm in the outer loop tries out a subset of the feasible parameter configurations to approximate the optimum. This concept goes by many different names, such as "simulation optimization" [27], "simulation evaluation" [28], "simulation integrated into optimization" [29], and "simulation-based optimization" [12]. If a combinatorial optimization problem is solved, in addition the term "simheuristic" has been coined [30]. If the simulation model is sufficiently complex, the optimization algorithm can only consist of general guidelines for searching good solutions. These general guidelines that are applicable across research domains are also referred to as meta-heuristics [31]. Meta-heuristics often start with some randomly drawn parameter configurations. After the first objective values are obtained, these values direct the further search. A good meta-heuristic balances exploration and exploitation. During exploration, parameter configurations quite different from the previous samples are tested. During exploitation, well-performing parameter configurations are slightly altered to obtain an improved parameter configuration. After several iterations, a stopping criterion is reached and the best solution so far found is returned as an approximation for the global optimum. For a guided search, a meta-heuristic needs to keep track of (a fragment of) the past evaluations. The vast amount of meta-heuristics stems from the fact that it is a non-trivial decision how to continue a search given a set of observations. Some meta-heuristics modify the best-performing parameter configurations directly and are therefore instance-based. Other meta-heuristics add a level of abstraction. They first derive a probability model from the previous observations and with the help of that probability model new parameter configurations are created.

Kotachi *et al.* [32] provide an example for an instance-based optimization. They simultaneously optimize the berth length, the amount of QCs, the number of gates, the fleet size of TTs, the number of import export rows and import rows, and the amount of RTGs per row. The objective function balances the throughput and the utilization weighted by investment costs. While a high throughput is achieved with more resources, the weighted utilization ensures that no superfluous resources are added to the container terminal. Even though not all permissible realistic values are taken into account, the authors calculate 72,576 possible parameter combinations. They build an optimization framework that consists of two stages: First, the interactions between the resources are examined to determine the most promising sequence of resources optimization tasks. As seven different resources are checked, 7! = 5040 possible permutations exist. Second, the gained sequence is utilized to optimize each resource one-by-one. The not yet optimized resources are selected according to stochastic sampling. To the best knowledge of the authors of the present publication, the aforementioned publication and this (including the prior study [19]) are the only ones that used the simulation-based optimization approach at a container terminal for scaling the amount of several utilized resources.

1.3. Relationship between Hyperparameter-Optimization and Simulation-based Optimization

Identifying high-performing solutions for a given model (in its abstract sense) is a typical research question in many fields of science. The necessary search process has been speeded up since the advent of computers and the new possibility to automate tedious and complex computations. Complex computations often turn out (a) to be quite resource intensive and (b) to often contain a large amount parameters that can be varied. Each parameter configuration describes an alternative shape of the executed computation. For categorical parameters (i.e. an element of a finite set), the size of the

[1] If the optimization problem (i.e. the objective values derived from the simulation model) has some known structural properties, one might prefer to derive a simpler representation that allows optimization with other tools [26].

 doi:10.20944/preprints202012.0194.v1

parameter configuration space grows exponentially with every additional parameter. For continuous parameters (e.g. a permissible range of real numbers), even for a single parameter the search space is infinite. Therefore, for many applications the parameter configuration reaches a size impossible or impractical to be covered exhaustively. Hence, scientists are forced to only evaluate a subset of all feasible parameter configurations. This search process is further complicated in case the model contains stochastic components. Hence, often a single parameter configuration needs to be tested several times before the result statistics reliably inform the scientist about the quality of a parameter configuration.

The machine learning community deals with learning algorithms that consist of many exchangeable components. For neural networks, e.g. for the activation function different mathematical functions can be inserted, the weights inside a neural network can be adjusted by different algorithms, the amount of neurons for each layer can vary etc. [33]. These decisions are referred to as hyper-parameters and are usually considered constant during one experiment. It is a non-trivial problem to identify the best hyper-parameters for a given machine learning problem. Since machine learning pipelines often contain stochastic components, a repeated evaluation is often necessary.

The task of constructing and adjusting machine learning is so complex that in some cases randomly picking parameter configurations outperformed the manual model calibration by scientists [34]. The authors explain these results with the higher resolution of the search grid and less wasted computational budget on the variation of parameters that have little or no impact on the final result. For supporting the expert in automating the search through a parameter configuration space, Hutter *et al.* [35] were the first to present an optimization procedure which can deal with numerical and categorical parameters in a problem-independent way. Bergstra *et al.* [36] quickly followed with an alternative approach and called it TPE. A comparison between several data-sets showed that the performance of such a hyper-parameter optimization technique varies with each setup [37–39]. The research topic is often referred to as hyper-parameter optimization and is subject to active research and development [39–41].

In the past years, many newly-developed optimization approaches have pushed the field forward. Comparison studies such as [37–39] are very crucial to identify strengths and weaknesses of these approaches. Sörensen [42] admonishes that the design and application of new meta-heuristics should go beyond playing a simple *up-the-wall game*. When meta-heuristics are designed and then applied to a few problems of the researchers' choice, this provided little generalizable knowledge. This topic is very much intertwined with the No Free Lunch Theorems (NFLT) in the optimization of black-box functions: One cannot determine the most successful optimization algorithm for an unseen problem [43]. For simulation-based optimization, this means that a meta-heuristic might fail for a new simulation model. In machine learning, this means that the hyper-parameter optimization algorithm might fail for a new learning algorithm and/or a new data-set. This claim might be not in harmony with observations from scientific literature: often some algorithms tend to provide better results than others [44]. Therefore, no published ranking of different optimization algorithms is guaranteed to be reproducible for other problem instances. Yet when several optimization studies are jointly considered, the observed characteristics, strengths, and weaknesses of each optimization algorithm (e.g. a meta-heuristic) should fit into the broader picture. Hence, a comparison study like this publication contributes to these deeper insights.

1.4. Integrating Decisions using Simulation-based Optimization

This study uses simulation-based optimization on a multivariate optimization problem at a container terminal. As parameters, only categorical, discrete, and continuous value ranges are permissible. This makes it suitable for choosing policies, determining the amount of employed equipment, and tuning policies that accept parameters. The simulation model is treated as a black-box function that is repeatedly evaluated due to its stochasticity. The novelty of this optimization study is that meta-heuristics, one of them previously developed in the context of hyper-parameter optimization

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 December 2020 doi:10.20944/preprints202012.0194.v1

in machine learning, is applied to a discrete event simulation model that models several integrated problems of a container terminal. As only meta-heuristics are deployed, the parameter configuration space and the simulation model can both be extended to represent more complex integrated decisions with little effort. Other approaches that couple simulation and optimization require to keep both a simulation model and a mathematical model aligned [29]. This study examines how reliably a meta-heuristic detects good parameter configurations.

2. Materials and Methods

First the simulation model is presented in Subsection 2.1, then the later used meta-heuristics are presented in Subsection 2.2. In Subsection 2.3 it is presented how the meta-heuristics are used for optimizing the simulation model at hand.

2.1. Simulation Model

In the following, the created simulation model of the container terminal and its restrictions are presented. The implementation takes place in Tecnomatix Plant Simulation and is based on the data of a real terminal. The layout of the terminal is shown in Figure 2.

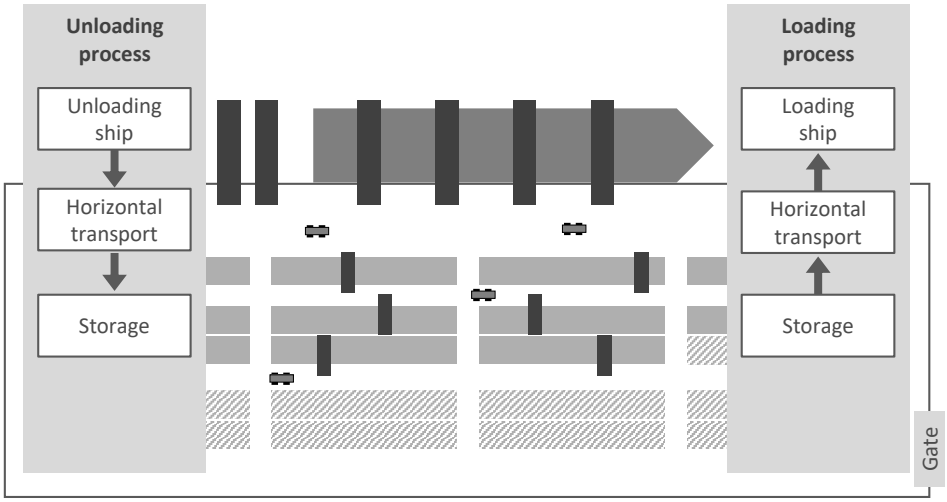


Figure 2. Layout and process illustration of the simulation model

The terminal has a quay length of 800 meters and a total of 20 YBs. Thereby, the designed simulation model shows the container handling between quayside and the container yard. As displayed in Figure 2, details on the design of the land-side transport interface as well as the berths are not considered in detail. In the study, the parallel handling of several ships is not modeled. Consequently, the simulation model of the container terminal is stressed by the arrival of one ship. For this, 12 bays of the ship with a total of 4,000 containers have to be handled. About half of them are import containers to be unloaded and the other half of them are export containers to be loaded. The container transport orders are pre-defined in tables, which also contain details such as start and destination of the containers at the terminal and the availability time for handling on the yard or quayside by RTGs. To reduce the calculation time of the simulation model, three tables per scenario are generated before the experiments start. Depending on the experiment, at least three and at most six QCs are available for loading and unloading the ship. The ship is unloaded and loaded bay by bay, whereby the QC always handles the entire bay that is assigned to it. Thus, the QC moves after the unloading and following loading of one bay. These bay changes are represented by a delay between the handling of two containers of the same QC. The less QCs are activated, the more bays must be served by one QC.

In the simulation model, an average quayside handling rate of 30 containers per hour is assumed, which is represented by the handling frequency of containers. Hence, the expected value for the

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 December 2020 doi:10.20944/preprints202012.0194.v1

handling of containers by the QC is 120 seconds. To model stochastic influences, a triangular distribution of the QC handling times is assumed with a minimum handling time of 80 seconds and a maximum value of 180 seconds. TTs carry out horizontal transport between the quay and the yard. Therefore, intermediate storage of the containers on the quay is not possible and the QCs have to load the containers directly onto the TTs. The travel times of the TTs are determined by the distances between QCs and YBs as specified by the terminal layout. For calculating the travel times, an average speed of the TTs of 8.4 m/s is assumed. By inserting a triangular distribution, stochastic influences are taken into account when calculating the travel times. The total number of TTs used per experiment is generated in the yard at the beginning of the simulation.

The inbound and outbound yard operations are performed by RTGs. For the simulation model, it is assumed that RTGs can perform an average of 15 containers handles per hour which corresponds to an expectation value of 240 seconds per handling. Deviations and irregularities in the process are modeled by a triangular distribution with a minimum handling time of 180 seconds and a maximum value of 420 seconds. For all experiments, at least two YBs are required for each active QC. Thus, it can be ensured that sufficient stowage space is available for the containers to be handled. Depending on the experiment, one of the two storage policies is investigated with the simulation model. For the first storage policies, containers from all active YBs can be transported to and from each of the QCs. In addition, import containers are randomly assigned to YBs, regardless of which QC is used for the unloading. Nevertheless, it is ensured that all active YBs are equally burdened as far as possible. The second storage strategy is the close assignment. This strategy seeks to minimize the distance between QCs and YBs. YBs with the shortest possible distance for horizontal transport are assigned to every QC. The handling of containers takes place between the defined QCs and YBs only.

Furthermore, two different dispatching policies are implemented in the simulation model. In the first strategy, a fixed number of TTs are assigned to each QC. This strategy is typically used in practical terminal operations as it is simplest to apply. Therefore, every TT gets an attribute which assigns it to a specific QC. In the second dispatching strategy, each TT can approach every QC. The strategy modifies the hybrid method from [9]. The choice of a next suitable job is based on the necessary driving time at the terminal and the waiting time of orders. The driving time and waiting time can be weighted differently for each experiment. For the selection of a suitable next order, each free TT inspects the next 20 orders. Now the order with the earliest availability time is determined. For the other 19 orders, the difference between the order's availability time and the earliest availability time is determined. Additionally, the required travel time to the start position of the respective order is calculated. Depending on the weighting factor of the experiment, both values are multiplied by the intended factor. Finally, the results are added up and the order with the smallest sum is chosen by the TT. If no TT is available at the quay, the QCs have to wait. Otherwise, the containers can be loaded directly onto the TTs. Loaded TTs drive the containers to the defined YB. There, the TTs and containers are separated from each other. Process steps for handling export containers can be taken over in the same way as described above, but in reverse order.

2.2. Employed Meta-Heuristics

Meta-heuristics are used to approximate the best parameter configuration for the given simulation model described in the previous subsection. The corresponding process is depicted in Figure 3. First, the history H and the counter i are initialized. Based on H, the meta-heuristic picks the experiment. In case of no prior experiments, usually parameters are selected randomly. In the next step, the meta-heuristic suggests a parameter configuration. If this is a new parameter configuration, a simulation experiment is executed and the fitness is calculated. Otherwise, from H the fitness value corresponding to $x^{(i)}$ is retrieved. In both cases, H is extended with the new value and then the meta-heuristic is setup with that H. After 50 evaluations, the results are reported and the optimization study is finished. The history H is implemented with a global database that even shares the history over several optimization runs independent from the employed meta-heuristic to reduce wall time.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 December 2020 doi:10.20944/preprints202012.0194.v1

Without having executed any simulation experiments, the simulation model must be considered a black-box. It is therefore impossible to know which of a set of given meta-heuristics would find the best approximation. Since many meta-heuristics themselves have a stochastic component, even two different optimization runs of the same meta-heuristic can result in different optima. Hence it is unpredictable whether for a yet unknown simulation model the optimum will be approximated sufficiently well and if so, which meta-heuristic will achieve this. Each meta-heuristic needs to prove its applicability to a problem empirically [44]. In this study, a TPE is applied to a discrete event simulation model and compared with Simulated Annealing (SA), BO, and Random Search (RS). These meta-heuristics are introduced next.

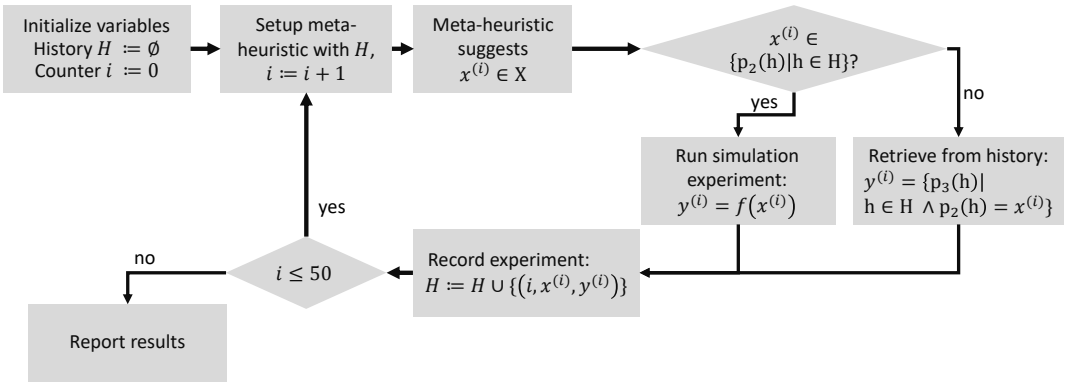


Figure 3. The optimization process

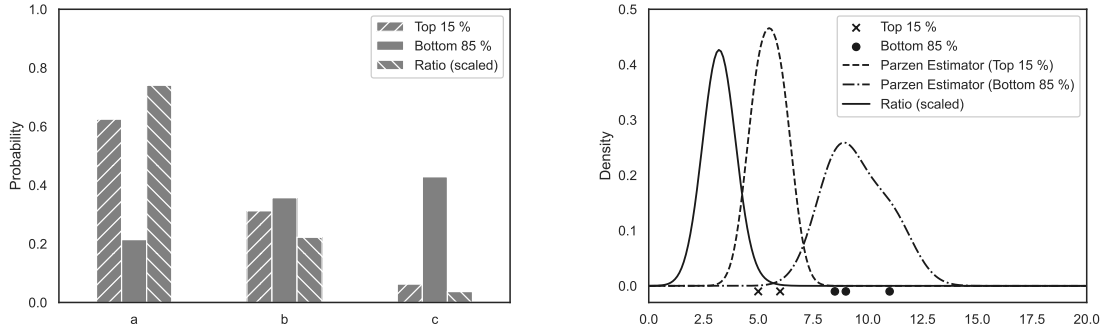
2.2.1. Tree-structured Parzen Estimator

TPE was developed to automate the search for a sufficiently well performing configuration of a Deep Belief Network [36]. For a Deep Belief Network, parameters were either categorical variables (e.g. the decision whether to use pre-processed data or use the raw data) or continuous variables (e.g. the learn rate). Integer values can either modeled as categorical variables (then the numbers are mere labels) or as continuous variables that are rounded before further use. The first approach is used e.g. for the amount of hidden units in the network and the second for choosing the batch size of either 20 or 100. In addition, dependencies between variables existed: One variable determined the amount of layers of the network and then each layer is configured on its own. Having the configuration of the third layer as part of a parameter configuration is only reasonable if the variable encoding the amount of layers is set to at least three. Such a parameter is called a conditional parameter. To reflect the dependencies, this kind of parameter configuration space is reasonably represented as a tree. This requires specific meta-heuristics that support such a tree structure and exploit its properties. The TPE approach has shown good benchmark results [37,45] and the initial paper is one of the major cited publications in hyper-parameter optimization in machine learning — at the time of writing, scopus indicates 939 citations.

TPE models $p(y < y*)$, $p(x|y < y*)$, and $p(x|y \geq y*)$, where p denotes a probability density function (short: density), y is a point evaluation of the model, and x is a parameter configuration. The first density $p(y < y*) = \gamma$ is a fixed value (e.g. 0.15 is used in the first publication) set by the experimenter and $y*$ is altered to fit the set value of γ for each iteration. The other two densities can be summarized as $p(x|y)$: Given a desired point evaluation value, what is the probability that a certain parameter configuration has been used. Commonly TPE is formulated to find a minimum and therefore $p(x|y < y*)$ describes the density of parameters which have shown better results whereas $p(x|y \geq y*)$ describes which parameters led to a lower performance. As the true probability density functions are unknown, they need to be estimated based on the obtained model evaluations at each iteration. For each categorical parameter, two probability vectors are maintained and updated: given the prior vector of probabilities $p = (p_1, ..., p_N)$, each probability p_i for $i \in 1, ..., N$ representing one category, the

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 December 2020 doi:10.20944/preprints202012.0194.v1

posterior vector elements are proportional to $N \cdot p_i + C_i$ where C_i counts the occurrences of choice i in the so far recorded model evaluations. An example is depicted in Figure 4a. The estimator for the better performing models (the Top 15 %) and the estimator for the worse performing models (the Bottom 85 %) are calculated based on the observations already recorded. For each continuous parameter, two adaptive Parzen estimators are used. Given a prior probability density distribution determined by the experimenter, with each point observation obtained from the model, the densities are further approximated. An example is depicted in Figure 4b. The parameter choices of the better and worse performing models are used to create the respective densities. A uniform prior distribution has been assumed. At each iteration the model consisting of the two densities is used to pick the next parameter configuration x to evaluate. To achieve this, several parameters x are sampled from the promising distribution $p(x|y < y*)$. The parameter configuration x with the highest expected improvement is picked. This criterion is positively correlated with the ratio $p(x|y < y*)/p(x|y \geq y*)$ [36]. In Figure 4, this is referred to as ratio. The criterion prefers the parameter configuration that has a high probability to lead to small evaluation values and a low probability to obtain large evaluation values for the minimization problem at hand. After the parameter configuration of the simulation model has been evaluated, the new results are incorporated into the model, i.e. the probability estimators. For this publication, the reference implementation [45] in the version 0.2.5 (the newest at the time of conducting the study) provided by the original authors has been chosen.



(a) Categorical Variable **(b)** Continuous Variable

Figure 4. The TPE makes uses the ratio of better- and worse-performing parameters to guide the search.

2.2.2. Simulated Annealing

SA is a meta-heuristic that is applicable both to combinatorial and multivariate problems [46]. The latter case is relevant for this optimization study. For this publication, the implementation of [45] in the version 0.2.5 (the newest at the time of conducting the study) is used. This version of SA works on the tree structure presented in Figure 5. The tree structure requires some specific adjustments documented in [47]. The default initialization values of the implementation have been used.

2.2.3. Bayesian Optimization

BO, also referred to as Gaussian process optimization, aims at minimizing of the expected deviation [48]. The response surface (i.e. the objective function values for given parameter configurations) is estimated including the uncertainty about the result. For this publication, the implementation from [49] in the version 1.2.6 (the newest at the time of conducting the study) is used. Since this procedure does not support tree-structured parameter configuration spaces, two simplifications are made: First, the number of TTs for the fixed assignment and the number of TTs for the global assignment are grouped into one parameter. This parameter ranges like in the tree branch of global assignment. If the fixed assignment is chosen, the number of trucks is divided by the amount of QCs. Second, the dispatching weight is always set, even if it is not applicable according the the tree-structured parameter configuration space and it is not interpreted by the simulation model. For

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 December 2020 doi:10.20944/preprints202012.0194.v1

initialization, for each optimization run five random samples are drawn before BO starts to guide the search.

2.2.4. Random Search

RS serves as a baseline. According to the NFLT, for some optimization problems meta-heuristics perform worse than RS. It is crucial to identify such meta-heuristics that misguide the search, e.g. by getting stuck in local optima. For this publication, the implementation of [45] in the version 0.2.5 (the newest at the time of conducting the study) is used. It is capable of sampling from a tree-structured tree-structured parameter configuration space.

2.3. The Optimization Problem

The optimization procedure is written as an external program that encapsulates the simulation. First, the simulation model is initialized with the parameter configuration to examine. After the simulation run is finished, the output of the simulation model is read. The communication between the two programs is realized through the COM-Interface. These values are inserted into the objective function that determines the fitness of a given solution. This optimization procedure is described in more detail in the following.

2.3.1. Parameter Configuration Space

During initialization, each parameter of the parameter configuration is set as a global variable in the simulation model. The interpretation of one global variable in the simulation model can depend on another global variable, e.g. the parameter *#TTs* is interpreted as the number of TTs for each QC in case of a fixed QC-TT assignment but interpreted as the number of TTs for all QCs in case of a free QC-TT assignment. The parameter *Dispatching Weight* is the weight for the travel time and ranges from 0 to 1 in steps of 0.1. The weight for the availability time for handling is deduced by calculating $1 -$ dispatching weight for travel time. This parameter is only used in case of a free QC-TT assignment. While it is not harmful for the simulation experiment to set that uninterpreted parameter anyways, the recorded observations for the meta-heuristic are flawed. This way a meta-heuristic might use the record including the uninterpreted parameter for guiding the further search despite the lack of any effect. This might guide the search process into a wrong direction.

In Figure 5 the parameter configuration space is depicted in its tree shape. It is dependent on the number of QCs which can be either 3, 4, 5, or 6. Each case is considered as an independent case that requires optimization.

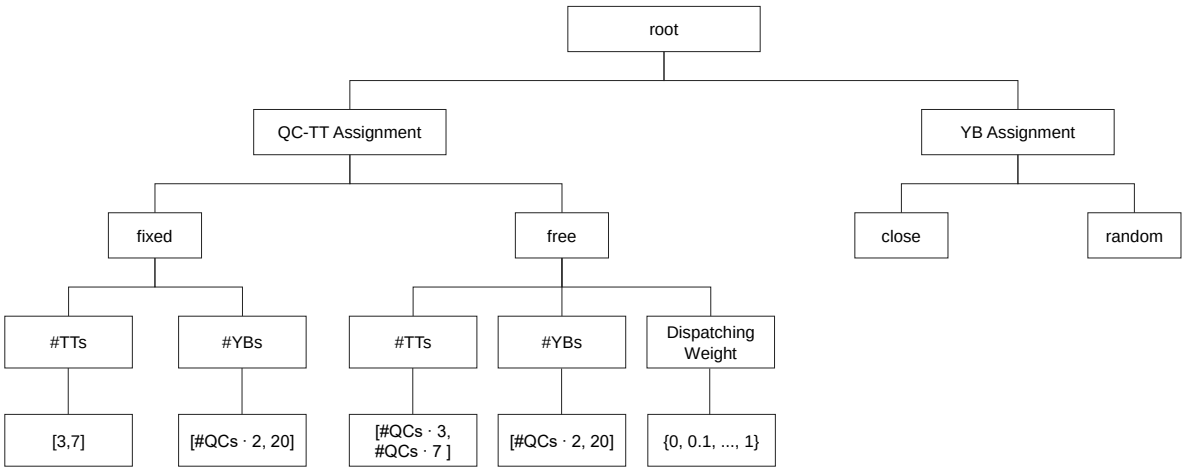


Figure 5. The parameter configuration space in tree shape

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 December 2020 doi:10.20944/preprints202012.0194.v1

The presented tree is used for TPE, SA, and RS. For BO, a vector representation is derived: Each of the parameters *QC-TT Assignment*, *YB Assignment*, *#TTs*, *#YBs*, and *Dispatching Weight* are represented by one dimension of this vector. The different interpretation of *#TTs* is alleviated by varying the value over the range $\#QCs \cdot 3$ to $\#QCs \cdot 7$ and in case of a fixed assignment dividing by *#QCs* and then rounding that value to the closest integer value before using it for parameterizing the simulation model. The parameter *Dispatching Weight* theoretically could take any value between 0 and 1. In order to make use of the caching mechanism that reduces wall time, only steps of 0.1 are permitted in this study.

2.3.2. Objective Function

After a simulation run is executed, the objective function is invoked to calculate the fitness for the given parameter configuration. The objective function needs to reflect the fact that the unloading and loading process needs to be fast while at the same time resources are only added if they are used later on. Therefore, the following objective function was developed (based on [32]):

$$fitness = \frac{\widetilde{t_{ship}}}{t_{ship}} \cdot \frac{50 \cdot \#QCs \cdot util_{QCs} + 5 \cdot \#RTGs \cdot util_{RTGs} + \#TTs \cdot util_{TTs}}{50 \cdot \#QCs + 5 \cdot \#RTGs + \#TTs} \tag{1}$$

The left factor of the multiplication reflects the inverted relative makespan of the ship. t_{ship} is the time used to unload and load the ship. As an approximate for $\widetilde{t_{ship}}$, prior to the optimization runs 100 random samples are drawn from the parameter configuration space and the makespan for the ship is measured. This normalization process centers the left factor around 1 and ensures that it stays in proportion to the right factor. The shorter the makespan, the larger the left factor turns out.

The right factor of the multiplication is the weighted utilization. *#QCs* refers to the amount of QCs, *#RTGs* the number of RTGs and therefore also the YBs, and *#TTs* the number of trucks. $util_{equipment}$ refers to the ratio the equipment has been working compared to the overall makespan. When summarizing these utilization values to one factor, weights are assigned according to the investment costs. It is assumed that a QC is fifty times more expensive than a truck and the cost of an RTG is 10 % of a QC [32]. The more equipment is utilized (with a special focus on expensive equipment), the larger the right factor turns out.

2.3.3. Structure of Optimization Study

For each scenario (3, 4, 5, and 6 QCs) and meta-heuristic (TPE, BO, SA, and RS), 50 optimization runs are executed. This allows to gain insights on the reproducibility of the optimization results using meta-heuristics. Each optimization run consists of 50 experiments. For each experiment, 30 simulation runs are executed. The results of each simulation run varies slightly due to stochastic factors. These are implemented by drawing handling times from random distributions as described in Subsection 2.1.

3. Results and Discussion

3.1. Preparatory study

The objective function (see Equation 1) requires $\widetilde{t_{ship}}$, the median of t_{ship}. The population parameter is only known after an exhaustive coverage of the parameter configuration space which must be avoided for an optimization study. As a replacement, an approximation must suffice. For that purpose, for 3 to 6 QCs in total 100 parameter configurations were sampled randomly and used to run the corresponding simulation experiments. For each of those simulation experiments, the makespan was recorded. The median, minimum, and maximum are noted in Table 1.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 December 2020 doi:10.20944/preprints202012.0194.v1

Table 1. Makespan for 100 randomly drawn experiments for each number of QCs (in hours, rounded).

Number QCs	Median	Minimum	Maximum
3	61	51	158
4	49	39	161
5	47	38	160
6	35	29	161

In Table 1 it is shown that the median and the minimum both decrease with an increasing amount of QCs. This is not true for the maximum which remains rather constant around 160 h. Furthermore, the difference between 4 and 5 QCs is comparably small. This can be explained by the fact that in the simulation model 12 bays of the ship always have to be handled. With 4 QCs, each QC handles exactly three bays. With 5 QCs, three QCs are responsible for two bays each and two QCs for three bays each. This means that three QCs finish the container handling earlier, but the makespan is based on the QC that finishes last.

3.2. Observations from all experiments

In the scope of the optimization study, in total 40,000 experiments are evaluated. For each scenario (3, 4, 5, and 6 QCs) and each meta-heuristic (TPE, BO, SA, and RS), fifty optimization runs are executed, each consisting of 50 experiments (see Figure 3). This set of experiments covers both many randomly chosen experiments (e.g. RS or initialization phase of any of the meta-heuristics) and experiments that are biased by the way each meta-heuristic works during its search process. This overview that leaves aside the search process provides some insights into the characteristics of the simulation model.

For each experiment, among others the makespan as well as the utilization of the equipment is measured. The utilization is the arithmetic mean of the working time of all equipment of its respective type. In Figure 6, the utilization of TTs, YBs, and QCs is shown. Due to the larger investment costs, the weighted utilization is closest to the QCs. The median of the utilization of the TTs is the lowest. Since a TT cannot lift a container itself, it must wait for a crane (either QC or RTG) to load or unload the TT. A high utilization of QCs and YBs is only possible if enough TTs are available which inevitably results into a lower utilization on the TT side. As the utilization of TTs is assigned a rather small weight in the fitness function, the lower utilization rate carries no (or more precise: less) weight.

In Figure 7, the difference between the maximum and minimum working time of the TTs for each experiment that used the global assignment policy is depicted. This is an indicator for how well the work could be shared among the TTs. As a general tendency it can be seen that more QCs positively correlate with a lower time difference.

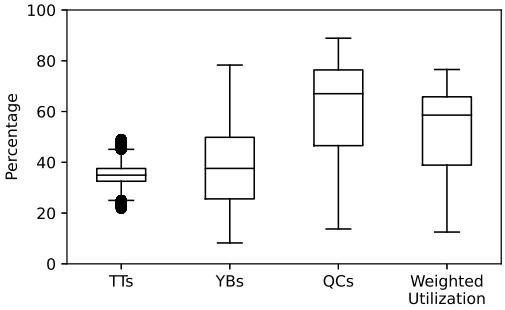


Figure 6. The utilization of the equipment over all experiments.

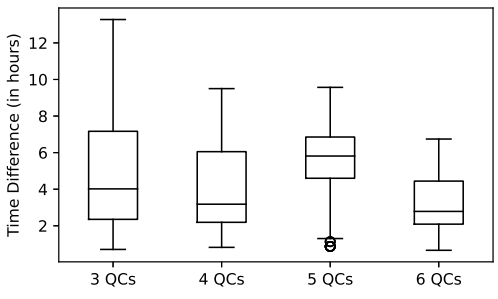


Figure 7. Time difference between maximum and minimum working time of the TTs.

 doi:10.20944/preprints202012.0194.v1

3.3. Approximated Optima

Both the preparatory study and the first screening of all executed experiments created first impressions of the underlying processes. Within the parameter configuration tree, there are exceptionally low performing solutions. This leaves the question: Which of the meta-heuristics identified the best parameter configuration(s)? For this, for each optimization run the experiment with the highest fitness is extracted. During optimization, both TPE and BO pick the next experiment with the highest expected improvement. Hence, unlike SA, after a phase of exploitation (i.e. minor adjustments) a phase of exploration (i.e. larger changes) can follow. RS is the most extrem example since exploitation is never soughted for. This in turn means that the best result of an optimization run can appear at any position of the sequence of recorded experiments.

In Figure 8, the four meta-heuristics TPE, BO, SA, and RS are compared for each scenario of using 3, 4, 5, and 6 QCs. Consistent with the preparatory study and for the same reasons, the results for 5 QCs are exceptionally worse compared to all other scenarios. Each of the meta-heuristics show outliers in at least three of the boxplots. This indicates the importance of stochastic influences during the search process. There is no clear ranking among the meta-heuristics. For 3 QCs, BO performs exceptionally worse than the other meta-heuristics, including RS. These results are repeated for 4 and 5 QCs, even though in less severity. This is especially interesting since for 6 QCs, BO shows the highest median. Over the first three scenarios, RS and SA show a very similar performance and for 6 QCs SA shows the worst median with outliers in both directions. The TPE performs very well on the first three scenarios, both providing many of the best solutions and little outliers that are substantially worse. For 6 QCs, the median of TPE is much lower than the media of BO. However, in two instances BO arrived at substantially lower performing solutions. The wide range of approximated optima and the large difference between the meta-heuristics are indicators for the complexities of the simulation model. The parameter configurations of all optima are closer examined in the following. The meta-heuristics always identifies that the fixed assignment of TTs to QCs provides an inferior performance compared to pooling. Furthermore, in all instances the pairing of each QC with its closest YBs performs better than if each container is delivered to a randomly chosen YB. Compared to these parameters, the dispatching shows no clearly interpretable results.

In Figure 9, the frequency of specific numbers of YBs for 3, 4, 5, and 6 QCs are depicted. A small number of YBs creates a bottleneck in the yard, while a too large number of YBs affect the result in two ways: First, the low utilization negatively affects the overall fitness directly. Second, TTs are able to combine transportation jobs in case they serve several QCs simultaneously. The less YBs are used, the shorter the traveled paths are and the higher the probability that an unloading job can be easily combined with a loading job. A typical pattern is that the number of used YBs often is a multitude of the number of QCs. This can be explained with the rather conservative transportation job assignment policy in place. This policy is designed to avoid traffic jam and rather postpones a job.

In Figure 10, the number of TTs per QC for each scenario is presented. The median for 4 QCs has been manually shifted slightly downwards for better visibility. Two interesting aspects of the simulation model can be detected here. First, due to the permissible value range and the low assigned weight of the number of TTs for the fitness calculation, here many outliers exist. For each QC, from 4 QCs to 6 QCs the median is decreasing. This can be explained by the higher chance that a TT can simultaneously support a loading and unloading process at different QCs. For 3 QCs, due to the amount of jobs within an equivalent time window, the probability for this kind of combination is lower.

In this study, like in [19], the TPE shows the most robust behavior. At none of the 200 optimization runs, the TPE returned the worst approximation for the optimum. This might be an interesting fact for future optimization studies that use a simulation model with longer computation time and/or a shorter time budget. Longer simulation time and strict time limitations might not allow to repeat optimization runs as often as it has been done in this study. Under that circumstances, it might be more important to obtain a reliable result than to obtain a better approximation but only with a lower probability.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 December 2020 doi:10.20944/preprints202012.0194.v1

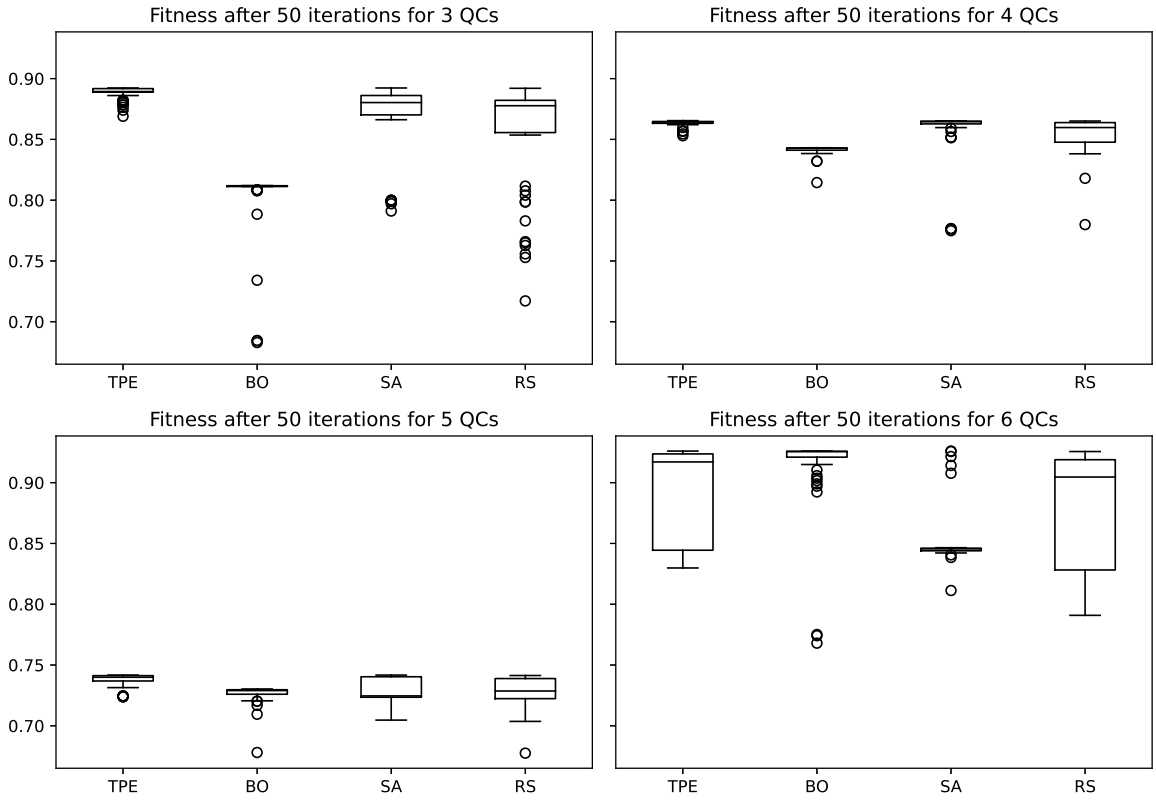


Figure 8. The approximated optima for each scenario and each meta-heuristic.

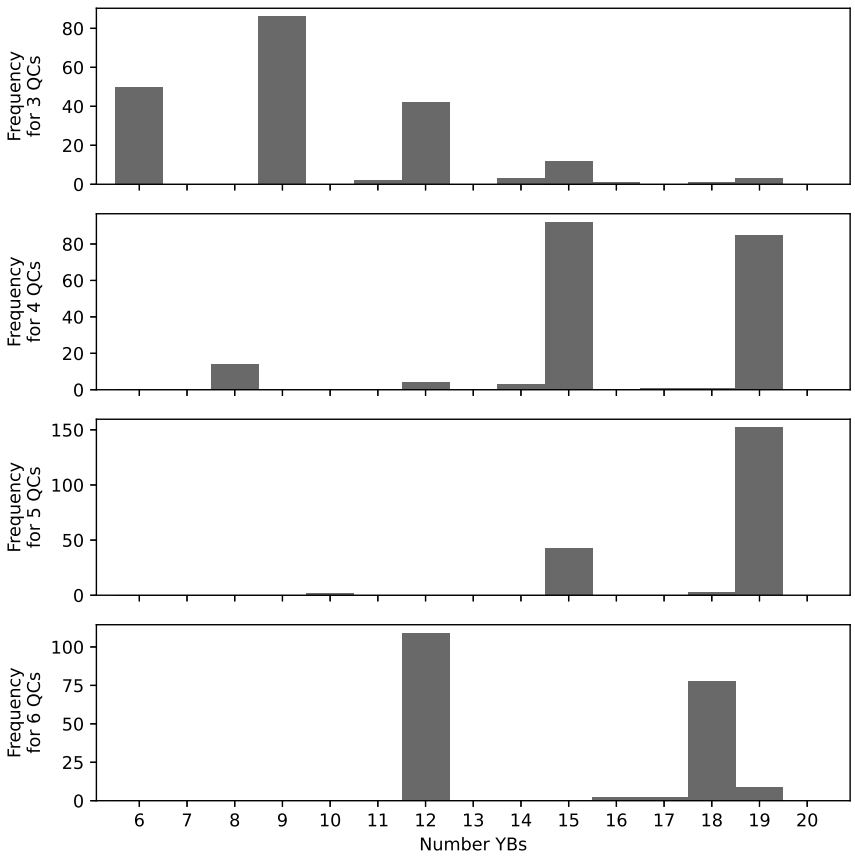


Figure 9. The number of YBs in the set of approximated optima.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 December 2020 doi:10.20944/preprints202012.0194.v1

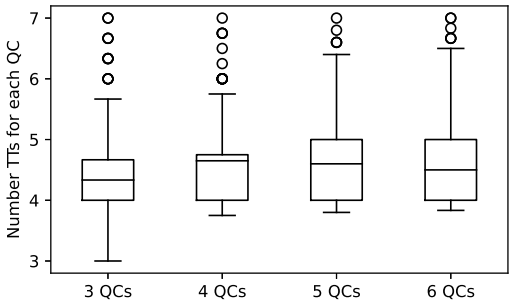


Figure 10. The number of TTs in the set of approximated optima.

In comparison to [32], this study shows an alternative approach to calibrate the number of equipment used in different functional areas of a container terminal. In addition, here policies are picked and tuned (if the policy accepts a parameter). The approach of Kotachi *et al.* [32] requires all parameters to be on an ordinal scale since they defined mutation as changing a parameter one level up or down. For categorical parameters that can take more than two values, there might be no intrinsic order. For continuous parameters, their method makes it necessary to define a gridsize. The approach presented in this study also converts the theoretically continuous parameter *Dispatching Weight* (ranging from 0 to 1) into a parameter that can take a value from the set $\{0, 0.1, ..., 1\}$. The purpose of this conversion is to avoid the evaluation of very similar parameter configurations – the already obtained experiment results are just loaded from a database. All used meta-heuristics support continuous parameters which might be of interest for future optimization studies.

4. Conclusions

This study provides an approach to solve integrated decision problems at container terminals. Earlier studies approach such problems often by a mathematical model that aims to optimize the schedule of jobs. Depending on the concept, sometimes the schedule is determined hours before the actual execution of a job. In rather deterministic environments, this is an appropriate approach. Another common approach in literature is to define a policy, i.e. a priority rule, that is evaluated using a simulation study. The design of experiments — e.g. a full-factorial design with a coarse grid — leads to either very large simulation studies or a selection of experiments biased by the researcher's beliefs. These shortcomings of optimization alone and manually designed large simulation studies are partly overcome by the presented simulation-based optimization approach. This approach uses simulation to evaluate the quality of a given solution. Only thus, the dynamics of real systems can be properly represented. Simulation-based optimization provides a possibility to illustrate these dynamics and provide an approximated solution for the problem. It is the only combination of simulation and optimization that does not rely on maintaining an additional mathematical optimization model [29]. Therefore, further decision problems can be integrated into the simulation model and the parameter configuration space with little effort. The authors show that meta-heuristics that originated from the domain of machine learning or which have been successfully applied in that area are in fact transferable. They can be used to optimize discrete event simulation models. A special focus is laid on discrete and continuous parameters that are potentially interdependent. Several optimization runs guided by different meta-heuristics are executed. With a restricted computational budget, promising parameter configuration ranges are identified. This publication focuses on examining the results of the different optimization runs.

For the simulation study carried out, processes at the container terminal are simplified in some cases. This results in some limitations of the simulation model. The generated model does not simulate variations of the system load due to the arrival of different sized container ships at the quayside. Furthermore, the simulation model works with fixed lists, which define the availability time for

 doi:10.20944/preprints202012.0194.v1

handling the containers on the quay or in the yard. In the very beginning, three different lists are generated for each experiment, which are randomly selected before every simulation run. However, this limits the stochasticity. Restrictions also had to be taken into account in the yard and its hinterland connections. In the simulation study, each YB is operated by only one RTG for simplifying the process. On a real terminal, RTGs move between YBs. If many storage and/or retrieval operations are necessary at one YB, an RTG can be added. These need to coordinate their work due to the non-crossing constraint. Furthermore, trucks arriving in the hinterland, which also have to be operated by the RTGs, are not taken into account in the simulation study. However, Schwientek *et al.* [50] show that hinterland traffic negatively affect on the productivity of RTGs only if the share of hinterland exceeds 30 %.

Due to the NFLT it is not clear whether these empirical results can be generalized to future studies that use simulation-based optimization. The applicability of meta-heuristics such as BO or TPE needs to be shown by more optimization studies — probably with various simulation models, different objective functions, and additional meta-heuristics (or different fine-tuning of the same meta-heuristic) for comparison. TPE offers many more configuration options which are not used, e.g. a uniform prior distribution is chosen for all parameters. The search process can be guided by selecting different prior distributions that reflect the belief of the domain expert [36,45]. BO in its implementation from [49] allows several adjustments. The expected improvement is only one of many possible criteria to choose the next experiment. As the surrogate function, except the standard Gaussian process e.g. a random forest can be chosen. In this study, the parameter configuration space is modeled as a tree with some variables repeated in different branches of that tree. This parameter configuration space construction is guided by the assigned importance of each decision while unnecessary repetitions are avoided. A deeper variation of a tree that could have been modeled in a shallow manner are only deeper if more nodes to determine the same parameter are added. At each of the nodes that determine the same parameter according to some meta-heuristic, the next parameter to evaluate is suggested in an independent manner. This also means that these nodes that reflect the same parameter need to share the computational budget and each receives less samples. A deeper investigation could shed some light on the trade-off between a shallow tree that ignores dependencies while each parameter is estimated with more samples on the one hand and a deep tree that reflects dependencies in its tree structure while each at each node the meta-heuristic needs to guide the next step based on less samples. Such insights could guide the researchers in constructing well-suiting parameter configuration spaces. It shall be noted that the best-suiting optimization tree could have been built if all dependencies and interactions were previously known. At the same time, if all dependencies and interactions were known, there might have been no need for simulation-based optimization in the first place.

In industry, the operational questions tackled in this publication with simulation-based optimization are solved on a daily basis. The solutions are not necessarily optimal but the management in charge might have some key insights on typical interactions of parameters based on experience. These might be used for restricting and designing the parameter configuration space, adjusting the prior probabilities in case of TPE etc. Yet since the practitioners are not necessarily familiar with the employed methodology, they might not be able to execute these tasks themselves. It is an open research area how to incorporate the experience of practitioners into the formalized optimization process of a simulation model.

Author Contributions: Conceptualization, M.K., A.S., C.J., and N.N.; methodology, M.K., A.S., and N.N.; software, M.K., A.S., and N.N.; validation, M.K.; formal analysis, M.K.; investigation, N.N., M.K.; resources, N.N.; data curation, M.K.; writing–original draft preparation, M.K., A.S., and N.N.; writing–review and editing, A.S., C.J.; visualization, N.N., M.K.; supervision, C.J.; project administration, M.K.; All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

BO	Bayesian Optimization
NFLT	No Free Lunch Theorem
QC	Quay Crane
RS	Random Search
RTG	Rubber-tired Gantry Crane
SA	Simulated Annealing
TPE	Tree-structured Parzen Estimator
TT	Terminal Truck
TEU	Twenty-foot Equivalent Unit
YB	Yard Block

References

1. UNCTAD. *Review of Maritime Transport*; 2020.
2. Gharehgozli, A.; Zaerpour, N.; de Koster, R. Container terminal layout design: transition and future. *Maritime Economics and Logistics* **2020**, *22*, 610–639. doi:10.1057/s41278-019-00131-9.
3. Bierwirth, C.; Meisel, F. A follow-up survey of berth allocation and quay crane scheduling problems in container terminals. *European Journal of Operational Research* **2015**, *244*, 675–689. doi:10.1016/j.ejor.2014.12.030.
4. Kizilay, D.; Eliiyi, D.T. A comprehensive review of quay crane scheduling, yard operations and integrations thereof in container terminals. *Flexible Services and Manufacturing Journal* **2020**. doi:10.1007/s10696-020-09385-5.
5. Chen, L.; Langevin, A.; Lu, Z. Integrated scheduling of crane handling and truck transportation in a maritime container terminal. *European Journal of Operational Research* **2013**, *225*, 142–152. doi:10.1016/j.ejor.2012.09.019.
6. Lange, A.K.; Schwientek, A.K.; Jahn, C. Reducing truck congestion at ports – classification and trends. Digitalization in Maritime and Sustainable Logistics; Jahn, C.; Kersten, W.; Ringle, C.M., Eds.; epubli: Berlin, 2017; Proceedings of the Hamburg International Conference of Logistics (HICL), pp. 37–58. doi:10.15480/882.1484.
7. Dragovic, B.; Tzannatos, E.; Park, N.K. Simulation modelling in ports and container terminals: literature overview and analysis by research field, application area and tool. *Flexible Services and Manufacturing Journal* **2017**, *29*, 4–34. doi:10.1007/s10696-016-9239-5.
8. Angeloudis, P.; Bell, M.G. A review of container terminal simulation models. *Maritime Policy & Management* **2011**, *38*, 523–540. doi:10.1080/03088839.2011.597448.
9. Schwientek, A.K.; Lange, A.K.; Jahn, C. Effects of Terminal Size, Yard Block Assignment, and Dispatching Methods on Container Terminal Performance. In *Winter Simulation Conference 2020*; Bae, K.H.; Feng, B.; Kim, S.; S., L.M.; Zheng, Z.; Roeder, T.; Thiesing, R., Eds.; 2020.
10. Zhen, L.; Yu, S.; Wang, S.; Sun, Z. Scheduling quay cranes and yard trucks for unloading operations in container ports. *Annals of Operations Research* **2016**, *122*, 21. doi:10.1007/s10479-016-2335-9.
11. Kastner, M.; Pache, H.; Jahn, C. Simulation-based optimization at container terminals: a literature review. Digital transformation in maritime and city logistics; Jahn, C.; Kersten, W.; Ringle, C.M., Eds.; epubli GmbH: Berlin, 2019; Proceedings of the Hamburg International Conference of Logistics (HICL), pp. 111–135. doi:10.15480/882.2493.
12. Zhou, C.; Li, H.; Liu, W.; Stephen, A.; Lee, L.H.; Peng Chew, E. Challenges and opportunities in integration of simulation and optimization in maritime logistics. Proceedings of the 2018 Winter Simulation Conference; Rabe, M.; Juan, A.A.; Mustafee, N.; Skoogh, A.; Jain, S.; Johansson, B., Eds. IEEE, 2018, pp. 2897–2908. doi:10.1109/WSC.2018.8632202.
13. He, J.; Huang, Y.; Yan, W.; Wang, S. Integrated internal truck, yard crane and quay crane scheduling in a container terminal considering energy consumption. *Expert Systems with Applications* **2015**, *42*, 2464–2487. doi:10.1016/j.eswa.2014.11.016.

14. Cao, P.; Jiang, G.; Huang, S.; Ma, L. Integrated simulation and optimisation of scheduling yard crane and yard truck in loading operation. *International Journal of Shipping and Transport Logistics* **2020**, *12*, 230–250. doi:10.1504/IJSTL.2020.107234.
15. Castilla-Rodríguez, I.; Expósito-Izquierdo, C.; Melián-Batista, B.; Aguilar, R.M.; Moreno-Vega, J.M. Simulation-optimization for the management of the transshipment operations at maritime container terminals. *Expert Systems with Applications* **2020**, *139*. doi:10.1016/j.eswa.2019.112852.
16. Legato, P.; Mazza, R.M.; Trunfio, R. Simulation-based optimization for discharge/loading operations at a maritime container terminal. *OR Spectrum* **2010**, *32*, 543–567. doi:10.1007/s00291-010-0207-2.
17. Kizilay, D.; Eliiyi, D.T.; van Hentenryck, P. *Constraint and mathematical programming models for integrated port container terminal operations*; Vol. 10848 LNCS, *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2018. doi:10.1007/978-3-319-93031-2_25.
18. Sislioglu, M.; Celik, M.; Ozkaynak, S. A simulation model proposal to improve the productivity of container terminal operations through investment alternatives. *Maritime Policy and Management* **2019**, *46*, 156–177. doi:10.1080/03088839.2018.1481544.
19. Kastner, M.; Nellen, N.; Jahn, C. Model-based Optimisation with Tree-structured Parzen Estimation for Container Terminals. ASIM 2019 Simulation in Produktion und Logistik 2019; Putz, M.; Schlegel, A., Eds. Wissenschaftliche Scripten, 2019, ASIM-Mitteilung, pp. 489–498.
20. Singgih, I.K.; Jin, X.; Hong, S.; Kim, K.H. Architectural Design of Terminal Operating System for a Container Terminal Based on a New Concept. *Industrial Engineering and Management Systems* **2016**, *15*, 278–288. doi:10.7232/iems.2016.15.3.278.
21. Schwientek, A. Abilities of the Used Terminal Operating Systems: Personal conversation, 2012-2013.
22. Barton, R.R. Simulation experiment design. Proceedings of the 2010 Winter Simulation Conference; IEEE: Piscataway, N.J., 2010; pp. 75–86. doi:10.1109/WSC.2010.5679171.
23. Li, H.; Zhou, C.; Lee, B.K.; Lee, L.H.; Chew, E.P.; Goh, R.S.M. Capacity planning for mega container terminals with multi-objective and multi-fidelity simulation optimization. *IISE Transactions* **2017**, *49*, 849–862. doi:10.1080/24725854.2017.1318229.
24. Al-Salem, M.; Almomani, M.; Alrefaei, M.; Diabat, A. On the optimal computing budget allocation problem for large scale simulation optimization. *Simulation Modelling Practice and Theory* **2017**, *71*, 149–159. doi:10.1016/j.simpat.2016.05.004.
25. Ho, Y.C.; Zhao, Q.C.; Jia, Q.S., Eds. *Ordinal optimization: Soft optimization for hard problems*; Springer: New York, NY, 2007.
26. Xu, J.; Huang, E.; Chen, C.H.; Lee, L.H. Simulation optimization: A review and exploration in the new era of cloud computing and big data. *Asia-Pacific Journal of Operational Research* **2015**, *32*, 1–34. doi:10.1142/S0217595915500190.
27. Fu, M.C.; Glover, F.W.; April, J. Simulation optimization: a review, new developments, and applications. Proceedings of the 2005 Winter Simulation Conference; Kuhl, M.; Steiger, N.; Armstrong, F.; Joines, J., Eds. IEEE, 2005, pp. 351–380.
28. Figueira, G.; Almada-Lobo, B. Hybrid simulation–optimization methods: A taxonomy and discussion. *Simulation Modelling Practice and Theory* **2014**, *46*, 118–134. doi:10.1016/j.simpat.2014.03.007.
29. Hanschke, T.; Krug, W.; Nickel, S.; Zisgen, H. VDI 3633 Blatt 12 - Simulation und Optimierung. In *VDI-Handbuch Fabrikplanung und -betrieb - Band 2: Modellierung und Simulation*; Beuth Verlag: Berlin, 2016; Vol. 2.
30. Juan, A.A.; Faulin, J.; Grasman, S.E.; Rabe, M.; Figueira, G. A review of simheuristics: Extending metaheuristics to deal with stochastic combinatorial optimization problems. *Operations Research Perspectives* **2015**, *2*, 62–72. doi:10.1016/j.orp.2015.03.001.
31. Chopard, B.; Tomassini, M. *An introduction to metaheuristics for optimization*, 1st edition 2018 ed.; Natural Computing Series, Springer International Publishing: Cham, 2018.
32. Kotachi, M.; Rabadi, G.; Seck, M.; Msakni, M.K.; Al-Salem, M.; Diabat, A. Sequence-based simulation optimization: an application to container terminals. 2018 IEEE Technology & Engineering Management Conference; IEEE: Piscataway, NJ, 2018; pp. 1–7. doi:10.1109/TEMSCON.2018.8488396.
33. Géron, A. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems*, 2. ed.; O'Reilly UK Ltd., 2019.

34. Bergstra, J.; Bengio, Y. Random search for hyper-parameter optimization. *Journal of Machine Learning Research* **2012**, *13*, 281–305.
35. Hutter, F.; Hoos, H.H.; Leyton-Brown, K. Sequential model-based optimization for general algorithm configuration. Learning and intelligent optimization; Coello, C.A.C., Ed. Springer Berlin Heidelberg, 2011, pp. 507–523.
36. Bergstra, J.; Bardenet, R.; Bengio, Y.; Kégl, B. Algorithms for hyper-parameter optimization. 25th Annual Conference on Neural Information Processing Systems; J. Shawe-Taylor.; R.S. Zemel.; P. Bartlett.; F. Pereira.; K.Q. Weinberger., Eds.; Neural Information Processing Systems Foundation: Granada, Spain, 2011; Vol. 24, *Advances in Neural Information Processing Systems*, pp. 2546–2554.
37. Eggensperger, K.; Feurer, M.; Hutter, F.; Bergstra, J.; Snoek, J.; Hoos, H.; Leyton-Brown, K. Towards an empirical foundation for assessing Bayesian optimization of hyperparameters. NIPS workshop on Bayesian optimization in theory and practice, 2013, Vol. 10, pp. 1–5.
38. Madrigal, F.; Maurice, C.; Lerasle, F. Hyper-parameter optimization tools comparison for multiple object tracking applications. *Machine Vision and Applications* **2019**, *30*, 269–289. doi:10.1007/s00138-018-0984-1.
39. Yang, L.; Shami, A. On hyperparameter optimization of machine learning algorithms: Theory and practice. *Neurocomputing* **2020**, *415*, 295–316. doi:10.1016/j.neucom.2020.07.061.
40. Gijsbers, P.; LeDell, E.; Thomas, J.; Poirier, S.; Bischl, B.; Vanschoren, J. *An Open Source AutoML Benchmark*; 2019.
41. Akiba, T.; Sano, S.; Yanase, T.; Ohta, T.; Koyama, M. Optuna: A Next-generation Hyperparameter Optimization Framework. Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining; Teredesai, A., Ed.; Association for Computing Machinery: New York,NY,United States, 2019; ACM Digital Library, pp. 2623–2631. doi:10.1145/3292500.3330701.
42. Sörensen, K. Metaheuristics-the metaphor exposed. *International Transactions in Operational Research* **2015**, *22*, 3–18. doi:10.1111/itor.12001.
43. Wolpert, D.H.; Macready, W.G. No free lunch theorems for optimization. *IEEE transactions on evolutionary computation* **1997**, *1*, 67–82.
44. McDermott, J. When and why metaheuristics researchers can ignore "No Free Lunch" theorems. *Metaheuristics* **2019**, *1*, 67. doi:10.1007/s42257-019-00002-6.
45. Bergstra, J.; Yamins, D.; Cox, D.D. Making a science of model search: hyperparameter optimization in hundreds of dimensions for vision architectures. Proceedings of the 30th International Conference on Machine Learning; Dasgupta, S.; McAllester, D., Eds. JMLR.org, 2013, pp. 115–123.
46. Kirkpatrick, S.; Gelatt, C.D.; Vecchi, M.P. Optimization by simulated annealing. *Science (New York, N.Y.)* **1983**, *220*, 671–680. doi:10.1126/science.220.4598.671.
47. Bergstra, J. Simulated Annealing. https://github.com/hyperopt/hyperopt/blob/master/hyperopt/anneal.py, accessed on 29.11.2020.
48. Močkus, J. On Bayesian methods for seeking the extremum. Optimization techniques IFIP technical conference, 1975, pp. 400–404.
49. The GPyOpt authors. GPyOpt: A Bayesian Optimization framework in Python. http://github.com/SheffieldML/GPyOpt, accessed on 29.11.2020.
50. Schwientek, A.; Lange, A.K.; Jahn, C. Simulation-based Analysis of Dispatching Methods on Seaport Container Terminals. ARGESIM Report AR 59; Deatcu, C.; Lückerath, D.; Ullrich, O.; Durak, U., Eds., 2020, pp. 357–364. doi:10.11128/arep.59.a59050.