Surrogate-based optimization of a periodic rescheduling algorithm

Ikonen, Teemu J.

2022-06

Ikonen , T J , Heljanko , K & Harjunkoski , I 2022 , ' Surrogate-based optimization of a periodic rescheduling algorithm ' , AIChE Journal , vol. 68 , no. 6 , 17656 , pp. e17656 . https://doi.org/10.1002/aic.17

http://hdl.handle.net/10138/344131
https://doi.org/10.1002/aic.17656

Received: 22 July 2021 | Revised: 7 February 2022 | Accepted: 14 February 2022

DOI: 10.1002/aic.17656

AIChE JOURNAL

RESEARCH ARTICLE

Process Systems Engineering

Surrogate-based optimization of a periodic rescheduling algorithm

Teemu J. Ikonen[1] | Keijo Heljanko[2,3] | Iiro Harjunkoski[1,4]

[1]Department of Chemical and Metallurgical Engineering, Aalto University, Aalto, Finland

[2]Department of Computer Science, University of Helsinki, Helsinki, Finland

[3]Helsinki Institute for Information Technology (HIIT), Helsinki, Finland

[4]Hitachi Energy Research, Mannheim, Germany

Correspondence
Teemu J. Ikonen, Department of Chemical and Metallurgical Engineering, Aalto University, PO Box 16100, 00076 Aalto, Finland.
Email: teemu.ikonen@aalto.fi

Funding information
Academy of Finland, Grant/Award Numbers: 330388, 336092

Abstract

Periodic rescheduling is an iterative method for real-time decision-making on industrial process operations. The design of such methods involves high-level when-to-schedule and how-to-schedule decisions, the optimal choices of which depend on the operating environment. The evaluation of the choices typically requires computationally costly simulation of the process, which—if not sufficiently efficient—may result in a failure to deploy the system in practice. We propose the continuous control parameter choices, such as the re-optimization frequency and horizon length, to be determined using surrogate-based optimization. We demonstrate the method on real-time rebalancing of a bike sharing system. Our results on three test cases indicate that the method is useful in reducing the computational cost of optimizing an online algorithm in comparison to the full factorial sampling.

KEYWORDS

kriging, online scheduling, re-optimization, rolling horizon, surrogate modeling

1 | INTRODUCTION

Process scheduling is operational decision-making that arises in many fields of process systems engineering (PSE), such as production of metals, petrochemicals, pulp and paper, and transportation of goods. The main decisions involve defining the set of tasks to be executed, their timing, and the allocated resources.[1] In offline scheduling, these decisions are determined prior to the beginning of the process operation based on the available information. Mathematical optimization methods can find the optimal solution to a vast variety of such (often combinatorially complex) scheduling problems—subject to sufficient computing resources and execution time. For reviews of the developed optimization methods, we refer the reader to articles.[1–3]

In online scheduling, the scheduling decisions are made in real-time while operating the process. If the decision-making is based on mathematical optimization, this process typically involves iterative execution of optimization procedures, the results of which are used to update the current schedule. Each optimization procedure aims to find the optimal operations for a predefined scheduling horizon to the future using the latest information of the process. As this scheduling horizon moves ahead in time, the method is generally referred to as the rolling horizon approach.[2] While the methods for offline scheduling are fairly established, less attention has been put on the design of online scheduling methods.[4] A major challenge in online scheduling is how to optimally use new information and the given computing resources in a limited time. The high-level decisions related to online scheduling can be categorized into when-to-schedule and how-to-schedule decisions.[5] Gupta et al.[6] review the literature of online scheduling and highlighted four aspects for the design of online scheduling algorithms: (i) re-computation trigger, (ii) computation technology, (iii) uncertainty modeling, and (iv) allowable changes and constraints to the schedule. While the re-computation trigger is essentially a synonym for the when-to-schedule decision, the last three aspects are all different subcategories of how-to-schedule decisions.

Regarding the when-to-schedule decision, periodic rescheduling is perhaps the most commonly used in the literature to trigger scheduling procedures. Recently, Gupta and Maravelias[7] investigate how the re-optimization interval and horizon length of open-loop scheduling affect the closed-loop solutions. The same authors also propose a systematic way to design a periodic rescheduling algorithm,[4] focusing on the interval of scheduling procedures and the horizon length. Stevenson et al.[8] study the effect of periodic rescheduling interval to the performance of a multi-purpose production plant in different operation conditions. Dong et al.[9] present a re-optimization framework for a maritime inventory routing problem based on a discrete-time arc-flow formulation and stochastic simulations to account for uncertainty. Other commonly used methods for rescheduling timing are event-driven rescheduling[10,11] and a hybrid of periodic and event-driven rescheduling.[12] Ikonen et al.[13] propose an approach where a reinforcement learning agent is trained to decide on the timing and computing time allocation of rescheduling procedures. Recent studies on allowable changes and constraints in online scheduling are published in articles.[14,15] Vieira et al.[16] provide a thorough review of rescheduling strategies, including re-optimization triggers. In this study, we focus on the decisions related to the periodic rescheduling.

It is worth noticing that an evaluation of an online scheduling algorithm requires repeatedly performed optimization procedures, as well as a simulation of the process, to which the optimized scheduling decisions are applied. In order to obtain a reliable metric of the closed loop performance of the algorithm, the evaluation may take hours of computing time. Considering that design of an online scheduling algorithm has many algorithmic settings (discussed above), a full factorial evaluation of all settings may require a prohibitively long computing time. In the literature, this aspect of online scheduling is rarely discussed.

Surrogate-based optimization is a sample efficient global search method suitable for black-box type objective functions, the evaluation of which is computationally expensive. The main idea is to, first, generate an initial sampling plan of the search space based on a design of experiment method,[17] for example, the Latin hypercube sampling[18] or the Morris-Mitchell Maximin distance sampling.[19] Second, a surrogate (also referred to as a meta-model, a response surface, or an emulator) is constructed based on an initial sampling. The purpose of the surrogate is to serve as a cheap-to-evaluate representation of the objective function in the search space. Third, the surrogate is used in an iterative process where new candidate points (also referred to as infill points) are identified and evaluated, and the resulting input–output data are used to update the surrogate. A field partly overlaid with surrogate-based optimization is Bayesian optimization with the distinction that in the latter the surrogate is typically updated based on Bayesian statistics.[20]

Early works in these fields were done by Kushner,[21] Močkus and Žilinskas[22,23] during 1960–1970s. Sacks et al.[24] pioneered surrogate modeling of computational simulations in late 1980s. Jones et al.[25] proposed the efficient global optimization (EGO) method, where the infill points are determined based on a surrogate by maximizing the expected improvement in the objective function value (developing on the idea presented by Močkus[22]). Commonly used surrogates are radial basis functions,[26] kriging,[27] and polynomial functions. Surrogate-based optimization can take into consideration features such as noisy experiments,[28,29] multiobjective optimization,[30,31] multifidelity function evaluations,[32] and parallel evaluation of infill points.[33,34] Surrogate-based optimization, as well as Bayesian optimization, have been used extensively in the design of engineering systems.[35,36] McBride and Sundmacher[37] review surrogate modeling in chemical process engineering.

Regarding planning and scheduling, Wan et al.[38] deploy surrogate modeling in simulation-based optimization of a three-stage supply chain problem under uncertainty. Sahay and Ierapetritou[39] minimize the total cost of a multi-enterprise supply chain by finding the optimal warehouse capacities using surrogate-based optimization based on Kriging and expected improvement maximization. Shi and You[40] reformulate an integrated optimization problem as a mixed-integer nonlinear program and propose a bilevel structure, consisting of scheduling dynamic optimization levels. Surrogate models based on piecewise linear functions are used to represent the linking variables of the levels (i.e., processing times and costs). Candelieri et al.[41] apply Bayesian optimization to the scheduling of pump operations. Hutter et al.[42] developed Bayesian optimization algorithms for the optimal configuration of the mixed integer programming (MIP) solver CPLEX, often used to solve scheduling problems.

In this study, we first draw attention to the high computing cost of optimizing the closed-loop performance of an online scheduling algorithm via simulation. Second, we propose the optimization of continuous parameter of a periodic rescheduling algorithm to be performed using surrogate-based optimization, in order to mitigate the above-mentioned high computing cost. As far as we are able to ascertain, such an approach has not been reported in the literature. Third, Gupta and Maravelias[4] report results of a two-dimensional control parameter optimization using full factorial sampling, but, to our knowledge, results of systematic optimization in higher dimensions have not been reported in the literature. We extend the search space into three dimensions, and discuss how surrogate-based optimization could facilitate even higher dimensional search spaces.

We evaluate the approach on the bike sharing rebalancing problem (BRP), which is a generalization of the one-commodity pickup and delivery traveling salesman problem.[43] The objective is to minimize travel time of a number of rebalancing vehicles that transport bikes between fixed stations while fulfilling the service level requirement (SLR) of the system. We also discuss the transferability of the research to more complex processes in PSE, such as process scheduling and supply chain optimization. Artificial intelligence (AI) algorithms are used in this study (i) to make probabilistic predictions of the future demand at the bike stations, using a continuous time Markov chain, (ii) to cluster the stations of the system into subsets that are allocated to a single rebalancing vehicle, and (iii) in the surrogate-based optimization of control parameters of the online scheduling algorithm.

2 | OPTIMIZATION PROBLEM: BIKE SHARING REBALANCING

Urban bike sharing systems are expanding concurrently in large metropolitan areas and smaller cities around the world. In June 2021, the number of active bike sharing systems globally was around 2000.[44] Bike sharing is a form of sharing economy, aiming to increase the utilization rate of goods and reduce the overall greenhouse gas emissions.

The bikes of these systems are often used as last mile transportation, for example, between home and a train station. The operation of these systems is challenging, because the daily pickup and return distributions on different stations are asymmetric, leading to a risk of undesired empty or overfull stations. The system operators typically mitigate the imbalance by transporting bikes between stations by trucks. In the literature, the problem of finding the optimal operation of the rebalancing vehicles is referred to as the BRP.

The reason for selecting the bike sharing system as the example problem here is manifold. First, the process is very dynamic and has elements that follow certain patterns but cannot be fully forecasted. An example of these is customer behavior, which resembles the product demand in typical PSE applications. Second, both the dynamic BRP and a typical online production scheduling problem involve solving computationally expensive MIP models in limited amount of time. In addition, solving both problems follow similar procedures that need to be handled in the best possible manner, including the use of available data and information and an intelligent combination of optimization and AI technologies. Third, the studied problem is an instance of capacitated vehicle routing problems, which are frequently encountered by chemical companies that transport chemicals between and within their production sites, as well as to their customer network. Fourth, and perhaps the foremost, open data are widely available for the BRP and can be used for research and development purposes. This allows to develop, test, and publish methodologies in practice long before it would be possible with partially confidential company data. Instead of using anonymized data, we can directly refer to real process data collected from the described environment.

2.1 | Background

The problems studied in the literature can be categorized, in general, into static and dynamic BRPs. The static BRP corresponds to a case of overnight rebalancing, in which the effect of the users on the system during rebalancing is assumed to be negligible and the rebalancing vehicles can operate in the region without congestion. A typical objective is to minimize the cost of operations, subject to fulfillment of station-specific target levels. Raviv et al.[45] formulated MIP models to minimize a weighted objective function, combining the user dissatisfaction function[46] and total operating costs. Chemla et al.,[47] Dell'Amico et al.,[48] and Erdoğan et al.[49] formulate exact MIP models, and branch-and-cut algorithms to solve them, in order to obtain the optimal operations of rebalancing vehicles. Schuijbroek et al.[50] determine the SLRs from historical datasets and propose a clustering-based optimization approach. In their approach, each station is assigned to a self-sufficient cluster of stations, which is assigned a single truck to perform only transportations within the cluster. The operations of the vehicles are determined by a MIP model.

These models are useful in practice in the overnight rebalancing when the usage level is low, as shown, for example, by Freund et al.[51] in an experimental study in New York City. However, during the day-time, the state of the system is significantly altered by the users, leading to a situation where the optimal solution to the static BRP quickly becomes outdated. The expanded problem that also considers the usage of the system during rebalancing is referred to as the dynamic BRP. The key elements to solve this problem are as follows: (i) a good prediction of the future pickup and return rates at the stations (typically based on historical and online data), (ii) an efficient solution method (typically either MIP or a heuristic algorithm), and (iii) a re-optimization scheme that updates the planned operations when appropriate. Chiariotti et al.[52] propose a dynamic rebalancing approach that uses the birth-death process to model the future bike levels at the stations and updates the planned operations periodically using a heuristic algorithm. Ghosh et al.[53] propose a MIP model for the dynamic BRP that uses Lagrangian dual decomposition and an abstraction mechanism that groups stations close to each other into an abstract station in the optimization model. For an extensive review on BRPs, we refer the reader to the recent review paper by Shui and Szeto.[54]

2.2 | Real-time optimization problem

Let us now formally describe the real-time optimization problem of rebalancing a bike sharing system. The system consists of a set of stations $\mathcal{S}$ and a set of rebalancing vehicles $\mathcal{V}$ that perform the transportation of bikes in between the stations. Each station $i \in \mathcal{S}$ has C_i docks where bikes can be stored. Each vehicle $v \in \mathcal{V}$ has a maximum capacity of bikes, Q_v. The relative locations of the stations are characterized by the distance matrix $d_{i,j}$, where $i, j \in \mathcal{S}$. The vehicles are assumed to move from a station to another at a constant speed of $\bar{v}$. Loading a bike from a station to a vehicle, as well as the opposite action (i.e., unloading), takes time t_{load}. Initially, at time t_0, the stations have initial bike levels of s_i^{init}, $i \in \mathcal{S}$, and the vehicles initial loads of q_v^{init}, $v \in \mathcal{V}$, and initial locations of l_v^{init}, $v \in \mathcal{V}$, where l_v^{init} points to a station in set $\mathcal{S}$. A historical dataset of station-specific bike pickup $\mu_i(t)$, $i \in \mathcal{S}$ and return rates $\lambda_i(t)$, $i \in \mathcal{S}$ as a function of the time of the day t is available for the online scheduling algorithm*. Figure 1 shows these rates at a station located in a commercial district of Helsinki in June 2020.[55]

While also other re-optimization triggers are studied in the literature (see Section 1), we here only consider re-optimization procedures that are triggered periodically at an undefined time interval of Δt. We evaluate the periodic rescheduling algorithms by repeatedly formulating and

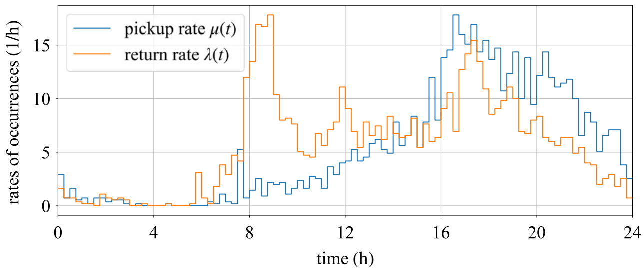


FIGURE 1 Average pickup and return rates, $\mu(t)$ and $\lambda(t)$, at a bike sharing station, located in a commercial district of Helsinki (Unioninkatu, 60.17°N, 24.95°E) in June 2020

solving optimization problems that determine the operations on the system (Figure 2). The re-optimization procedures are executed on a fixed-capacity computing resource that is continuously available. Thus, each optimization procedure is allocated a computing time limit equal to the re-optimization interval Δt. During the ith re-optimization procedure, the system evolves from the time point t_i to $t_{i+1} = t_i + \Delta t$. We simulate the evolution of the bike sharing system using real pickup and return data. The data used in the simulation is not included in the historical dataset. The operations obtained from the ith re-optimization procedure that started at t_i can be put into action at t_{i+1}. The information of bike pickups and returns by users during the evaluation becomes available for the online scheduling algorithm at the time they occur. We evaluate the algorithm during a time span of $[t_0 \ldots t_{\text{end}}]$. The objective is to minimize the sum of time the stations of the system are in an undesired state (i.e., empty or full), $\sum_{i \in \mathcal{S}} t_i^{\text{ef}}$, where t_i^{ef} is the time station i is empty or full. The next section describes the methods we use to formulate an optimization problem based on the prevailing state of the bike sharing system.

2.3 | Optimization problem formulation

We formulate the optimization problems using a slightly modified version of the model by Schuijbroek et al.[50] The model, in fact, consists of three sub-models: (i) a probabilistic prediction model of the future bike levels at the stations, (ii) MIP model to solve routing problems, and (iii) the aforementioned clustering model to decompose large problems into smaller ones. Although the model was originally designed for static BRP problems, the model is also well-suited for dynamic problems because it recognizes stations that do not need to be visited (referred to as self-sufficient stations) and it is scalable to larger problems due to its clustering feature that decomposes the original problem into smaller sub-problems. In the following, we describe the model and report our modifications to it. We follow mostly the symbol notation used by Schuijbroek et al.[50]

2.3.1 | Probabilistic prediction model

The inventory of bikes $S_i(t)$ at a station $i \in \mathcal{S}$ is modeled as a stochastic process on the state space $\{0,\ldots,C_i\}$, where C_i is the number of docks at the station (assuming that the station cannot be overfull). Let us denote this process as $\{S_i(t)\colon\ t \geq 0\}$, where t represents the time. Schuijbroek et al.[50] model this inventory as a $M_t/M_t/1/K$ queuing system,[56] such that the number of "customers" in the queue corresponds to the number of bikes at the station. The first two elements of the notation indicate that the arrivals of new "customers" (i.e., bikes in our case) and their service both are non-stationary Markovian processes. Both types of events are exponentially distributed with time-dependent rates $\lambda_i(t)$ and $\mu_i(t)$, respectively. The third element indicates that there is one machine serving the customers, and the fourth element that the queue has a total of $K = C_i$ waiting spaces. The queuing system is summarized in Figure 3.

A few assumptions need to be stated here (the same assumptions are made by Schuijbroek et al.[50]). First, the model assumes all bike users to travel alone. In reality, a portion of the users travel in batches and therefore arrive at the pickup and/or return stations nearly simultaneously, which is against the assumption of exponentially distributed bike pickups and returns (Poisson process). We estimate[†] that out of all trips conducted in Helsinki in June 2020 81.5% were made alone, 16.1% in pairs, and 2.4% in batches of three or more. As the majority of the trips are conducted alone, the effect of some users traveling in batches is presumably small. Second, the model considers the stations separately without considering possible interactions between them. As example of this, let us consider a user starting a trip who cannot pickup a bike from a station because it is empty (and the opposite case of ending a trip to a full station). These users would typically proceed next to a nearby station. Nevertheless, this behavior is not considered in the model.

Let us denote the transient probability for the number of bikes at station i to equal $\sigma \in \{0,\ldots,C_i\}$ at time $t \geq 0$ given that it was $s \in \{0,\ldots,C_i\}$ at time $t = 0$ as

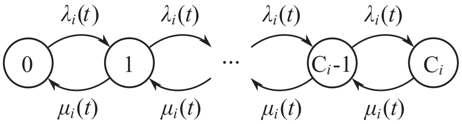


FIGURE 3 Markov chain of the bike inventory $S_i(t)$ at station $i \in \mathcal{S}$, having the capacity of C_i. The pickup and return rates of bikes are $\mu_i(t)$ and $\lambda_i(t)$, respectively

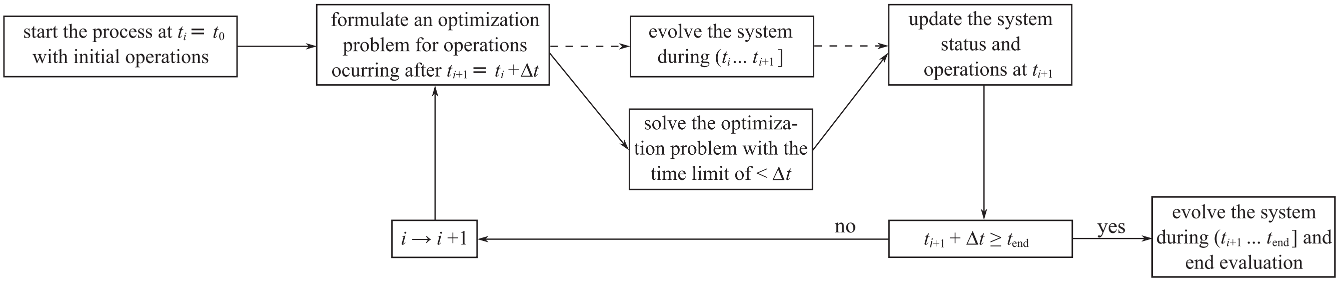

start the process at $t_i = t_0$ with initial operations
formulate an optimization problem for operations ocurring after $t_{i+1} = t_i + \Delta t$
evolve the system during $(t_i \ldots t_{i+1}]$
update the system status and operations at t_{i+1}
solve the optimization problem with the time limit of $< \Delta t$
$i \to i+1$
no
$t_{i+1} + \Delta t \geq t_{\text{end}}$
yes
evolve the system during $(t_{i+1} \ldots t_{\text{end}}]$ and end evaluation

FIGURE 2 Evaluation of a periodic rescheduling algorithm on an operating environment during a time span of $[t_0 \ldots t_{\text{end}}]$

$$p_i(s,\ \sigma,\ t)=\Pr(S_i(t)=\sigma|S_i(0)=s),\quad i\in\mathcal{S}. \tag{1}$$

Regarding the $M_t/M_t/1/K$ queuing system, no closed form solution exists for these probabilities. Nevertheless, we can solve the probabilities numerically using the Kolmogorov forward equations,[57] which for this system are as follows:

$$\begin{cases}\dot{p}(s,0,t)=\mu_i(t)p_i(s,1,t)-\lambda_i(t)p_i(s,0,t)\\ \dot{p}(s,\sigma,t)=\mu_i(t)p_i(s,\sigma-1,t)-[\mu_i(t)+\lambda_i(t)]p_i(s,\sigma,t)\\ \qquad\qquad +\lambda_i(t)p_i(s,\sigma+1,t),\quad \sigma\in\{1,\ldots,C_i-1\}\\ \dot{p}(s,C_i,t)=-\mu_i(t)p_i(s,C_i,t)+\lambda_i(t)p_i(s,C_i-1,t)\end{cases} \tag{2}$$

We use the Runge–Kutta method of order 5(4)[‡], implemented in the Python library Scipy,[58] to solve this system of differential equations.

Figure 4 shows two 4-h probabilistic predictions made at 6:00 and 14:00 for the Unioninkatu bike sharing station, based on the data shown in Figure 1, and the starting inventory at 6:00 and 14:00 on August 17, 2021. It is worth noticing that the station has higher return than pickup rate in the morning (Figure 1), and vice versa in the afternoon. Thus, the station is at risk of becoming full during the morning and empty during the afternoon. It is worth noticing that the prediction method is only based on historical data of pickups and returns on the station. The method does not consider other information (e.g., about the weather or nearby events), which could potentially improve the accuracy in practice.

From the operations point of view, we are interested in the fraction of demand that can be satisfied. Schuijbroek et al.[50] define the expected fraction of satisfied pickup and return demand at station i to be

$$\begin{cases}\dfrac{E[\text{satisfied bike pickup demands}]}{E[\text{Total bike pickup demands}]}=\dfrac{\int_0^T\mu_i(t)(1-p_i(s,0,t))dt}{\int_0^T\mu_i(t)dt}=g_i(s,0,T)\\ \dfrac{E[\text{satisfied bike return demands}]}{E[\text{Total bike return demands}]}=\dfrac{\int_0^T\lambda_i(t)(1-p_i(s,C_i,t))dt}{\int_0^T\lambda_i(t)dt}=g_i(s,C_i,T),\end{cases} \tag{3}$$

where T the length of the observation period. The authors define minimum values for these fractions, referred to as pickup and return SLRs, β_i^- and β_i^+, respectively, for all stations $i\in\mathcal{S}$. Regarding station i, let us denote its initial inventory by s_i^0 and the inventory after the rebalancing by s_i. Raviv et al.[46] provide a proof that $g_i(s,\ 0,\ T)$ and $g_i(s,\ C_i,\ T)$ increase and decrease, respectively, as the starting inventory s increases. Based on this proof, Schuijbroek et al.[50] determine the lower and upper bounds for the inventory s_i after rebalancing:

$$s_i^{\min}\le s_i\le s_i^{\max}, \tag{4}$$

where

$$\begin{cases}s_i^{\min}=\min\{s\in\{0,\ldots,C_i\}\ :\ g_i(s,0,T)\ge\beta_i^-\}\\ s_i^{\max}=\max\{s\in\{0,\ldots,C_i\}\ :\ g_i(s,C_i,T)\ge\beta_i^+\}.\end{cases} \tag{5}$$

The stations belonging to set $\mathcal{S}_0=\{i\in\mathcal{S}:s_i^{\min}\le s_i^0\le s_i^{\max}\}$ are referred to as self-sufficient stations. These stations do not require rebalancing, but can be used as a source or a sink for bikes (as long as the bounds are not violated). We make an assumption that satisfying the pickup and return demand is equally important and that all stations have the same importance[§]. This allows us to define a general system-wide SLR $\beta=\beta_i^-=\beta_i^+, i\in\mathcal{S}$.

If $s_i^{\max}\le s_i^{\min}$, $g_i(C_i,0,T)<\beta_i^-$, or $g_i(0,C_i,T)<\beta_i^+$, the lower and upper bounds cannot be satisfied. In such cases, we define the bounds to equal the inventory level that maximizes the worst of the two SLRs:

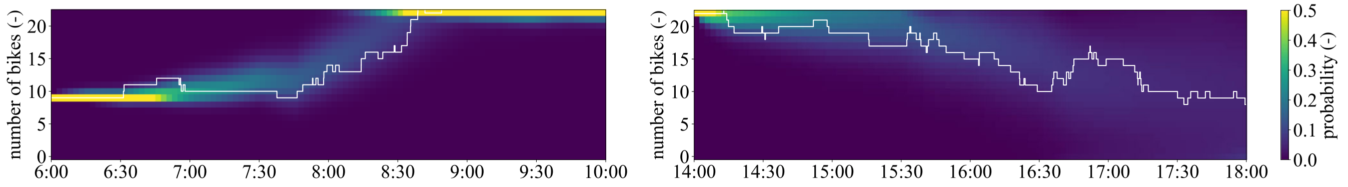
$$s_i^{\min}=s_i^{\max}=\underset{s\in\{0,\ldots,C_i\}}{\operatorname{argmax}}\{\min\{g_i(s,0,T),\ g_i(s,C_i,T)\}\}. \tag{6}$$

We define the set of stations to be visited initially as $\mathcal{S}^*=\mathcal{S}\setminus\mathcal{S}_0$. This set may, however, have a shortage of bikes or empty docks. In such a case, we incrementally add stations from $\mathcal{S}_0$ to $\mathcal{S}^*$ until $\mathcal{S}^*$ is self-sufficient or $\mathcal{S}^*=\mathcal{S}$. We add stations from $\mathcal{S}_0$ in the order of the greatest $g_i(s,\ 0,\ T)$ if the shortage is of bikes or $g_i(s,\ C_i,\ T)$ if the shortage is of docks (Equation 3).

Two focal control parameters of the probabilistic prediction method, presented in this section, are the length of the prediction horizon T and the SLR β. These control parameters, along with the re-optimization interval Δt, are optimized using a method, which we will describe later in Section 3.

2.3.2 | Routing model

The objective of the MIP routing model by Schuijbroek et al.[50] is to minimize the maximum travel distance among a set of rebalancing vehicles subject to satisfying the bounds in Equation 4 at all stations



(A) Morning

(B) Afternoon

FIGURE 4 Probabilistic prediction of the number of bikes at Unioninkatu bike station in Helsinki. The prediction is made based on historical data from June 2020 and given starting levels at 6:00 (A) and 14:00 (B) on August 17, 2020. The white lines show the actual realizations on that day. The station has a capacity of 22 bikes. The brightest yellow color exceeds the scale

$i \in \mathcal{S}$. Considering that the vehicles travel at the constant speed $\overline{v}$, the authors transform the time to pickup or deliver a bike into the corresponding distances d^{-} and d^{+}, respectively, which the vehicle would travel during these operations. These distances are included in the objective function. Therefore, the objective of the model is essentially to minimize the time to bring system to the above-described state that satisfies the SLRs. We have defined both the loading and unloading times to be equal to t_{load}, and thus in our case $d^{-} = d^{+} = \overline{v}t_{\text{load}}$.

The model, which is both arc- and sequence-indexed, has three sets of integer variables. Binary variable $x_{i,j,t,v}$ defines whether arc (i, j) is traveled at time step $t \in \mathcal{T}$ by the vehicle v. Integer variables $y^{-}_{i,t,v}$ and $y^{+}_{i,t,v}$ define the number of bikes pickups and deliveries, respectively, from/to station i at the time step t by vehicle v.

As we can see from the variable definitions, Schuijbroek et al.[50] define the model to take into account multiple vehicles[¶]. However, when they use it with the earlier mentioned clustering model, they solve the routing problem for one vehicle at the time. We use the routing model in the same way. Therefore, for us, set $\mathcal{V}$ has always the cardinality of 1. We use the following slightly modified version of the model:

$$\min H \tag{7}$$

$$s_i^0 + \sum_{t \in \mathcal{T}} \sum_{v \in \mathcal{V}} \left(y^{+}_{i,t,v} - y^{-}_{i,t,v} \right) \geq s_i^{\min}, \quad i \in \mathcal{S}^* \tag{8}$$

$$s_i^0 + \sum_{t \in \mathcal{T}} \sum_{v \in \mathcal{V}} \left(y^{+}_{i,t,v} - y^{-}_{i,t,v} \right) \leq s_i^{\max}, \quad i \in \mathcal{S}^* \tag{9}$$

$$\sum_{i \in \mathcal{S}^*} \sum_{j \in \mathcal{N}} x_{i,j,1,v} \leq 1, \quad v \in \mathcal{V} \tag{10}$$

$$\sum_{j \in \mathcal{N}} x_{i,j,t,v} \leq \sum_{j \in \mathcal{S}^*} x_{j,i,t-1,v}, \quad i \in \mathcal{S}^*, t \in \mathcal{T} \setminus \{1\}, v \in \mathcal{V} \tag{11}$$

$$\sum_{i \in \mathcal{S}^*} \sum_{t \in \mathcal{T}} \sum_{v \in \mathcal{V}} x_{i,i,t,v} = 0 \tag{12}$$

$$y^{-}_{i,t,v} \leq Q_v \sum_{j \in \mathcal{N}} x_{i,j,t,v}, \quad i \in \mathcal{S}^*, t \in \mathcal{T}, v \in \mathcal{V} \tag{13}$$

$$y^{+}_{i,t,v} \leq Q_v \sum_{j \in \mathcal{N}} x_{i,j,t,v}, \quad i \in \mathcal{S}^*, t \in \mathcal{T}, v \in \mathcal{V} \tag{14}$$

$$\sum_{t \in \mathcal{T}} \sum_{v \in \mathcal{V}} y^{-}_{i,t,v} \leq s_i^0, \quad i \in \mathcal{S}^* \tag{15}$$

$$\sum_{t \in \mathcal{T}} \sum_{v \in \mathcal{V}} y^{+}_{i,t,v} \leq C_i - s_i^0, \quad i \in \mathcal{S}^* \tag{16}$$

$$q_v^0 + \sum_{i \in \mathcal{S}^*} \sum_{\tilde{t} \in \mathcal{T}: \tilde{t} \leq t} \left(y^{-}_{i,\tilde{t},v} - y^{+}_{i,\tilde{t},v} \right) \geq 0, \quad t \in \mathcal{T}, v \in \mathcal{V} \tag{17}$$

$$q_v^0 + \sum_{i \in \mathcal{S}^*} \sum_{\tilde{t} \in \mathcal{T}: \tilde{t} \leq t} \left(y^{-}_{i,\tilde{t},v} - y^{+}_{i,\tilde{t},v} \right) \leq Q_v, \quad t \in \mathcal{T}, v \in \mathcal{V} \tag{18}$$

$$h_v = \sum_{i \in \mathcal{S}^*} \sum_{j \in \mathcal{S}^*} \sum_{t \in \mathcal{T}} d_{i,j} x_{i,j,t,v} + \sum_{i \in \mathcal{S}^*} \sum_{t \in \mathcal{T}} d^{-} y^{-}_{i,t,v} + \sum_{i \in \mathcal{S}^*} \sum_{t \in \mathcal{T}} d^{+} y^{+}_{i,t,v}, \quad v \in \mathcal{V} \tag{19}$$

$$H \geq h_v, \quad v \in \mathcal{V} \tag{20}$$

$$\sum_{j \in \mathcal{N}} x_{l_v,j,1,v} = 1, \quad v \in \mathcal{V} \tag{21}$$

$$\sum_{j \in \mathcal{N}} \sum_{t \in \mathcal{T}} x_{i,j,t,v} \leq 1, \quad i \in \mathcal{S}^*, v \in \mathcal{V} \tag{22}$$

$$\sum_{i \in \mathcal{S}^*} \sum_{t \in \mathcal{T}} x_{i,j,t,v} \leq 1, \quad j \in \mathcal{N}, v \in \mathcal{V} \tag{23}$$

$$\begin{aligned} & x_{i,j,t,v} \in \{0,1\}, \quad i \in \mathcal{S}^*, j \in \mathcal{N}, t \in \mathcal{T}, v \in \mathcal{V} \\ & y^{-}_{i,t,v}, y^{+}_{i,t,v} \in \mathbb{N}_0, \quad i \in \mathcal{S}^*, t \in \mathcal{T}, v \in \mathcal{V} \\ & H, h_v \geq 0, \quad v \in \mathcal{V} \end{aligned}$$

Constraints 8 and 9 impose that the SLR (specified by the inventory bounds in Equation 4) is fulfilled at all stations. Constraints 10–12 ensure that the route of vehicle v is started from one station, is continuous, and has no branching. Constraints 13 and 14 impose that the pickup and delivery of bikes to/from station i is only possible when leaving from it. The set $\mathcal{N} = \mathcal{S}^* \cup \{0\}$ includes an artificial station that allows operations to be performed at the final station of the route without incurring a new routing cost. Constraints 15 and 16 limit the amount of transshipment. Constraints 17 and 18 ensure that the load on vehicle v is within $\{0, \ldots, Q_v\}$ throughout the rebalancing. Constraint 19 tracks the total distance h_v traveled by vehicle v (taking also into account the bike pickup and delivery times). Constraint 20 tracks the maximum distance H among $h_v, v \in \mathcal{V}$.

Constraints 21–23 were not in the original model by Schuijbroek et al..[50] We add them, in order to make the model more suitable for dynamic BRPs. Constraint 21 imposes vehicle v to start the route from its starting location l_v. Constraints 22 and 23 enforce that a station can only be visited once, which reduces the computational complexity of the problem. Accordingly, in order to ensure the feasibility of the model, we adjust bounds $s_i^{\min}$ and $s_i^{\max}$ to be at most Q_v bikes away from initial number of bikes s_i^0 (i.e., the bounds can be satisfied by one visit). We reason this choice by the recursive nature of online scheduling. If a station requires multiple visits, a new visit would be scheduled at the next iteration after visiting the station for the first time. As all stations can be visited at most once we define the number of time steps in $\mathcal{T}$ to equal the number of stations in set $\mathcal{S}^*$.

The starting location l_v of vehicle v is its next destination after time Δt from the start of the re-optimization procedure. q_v^0 is the corresponding load when arriving to that station (i.e., the starting load for the next planned route).

Regarding Equation (14), Schuijbroek et al.[50] formulated it as

$$y^{+}_{i,t,v} \leq Q_v \sum_{j \in \mathcal{S}^*} x_{j,i,t,v}, \quad i \in \mathcal{S}^*, t \in T, v \in \mathcal{V} \tag{24}$$

In their formulation, the pickup of bikes can happen when leaving from a station, and delivery of bikes when arriving at a station. However, to our understanding, this form would allow the capacity of vehicles to be violated. In our experiments, we use Equation (14), defining that both pickup and delivery can happen only when leaving a station, precluding the violation.

When performing rebalancing with more than one vehicle, we first solve the clustering model[50] to decompose the problem into n single-vehicle routing problems (Equations 7–23), where n is the number of vehicles. The clustering model, which is also a MIP model, is defined in the paper by Shuijbroek et al.[50]

3 | SURROGATE-BASED OPTIMIZATION

In the previous section, we described the online algorithm for the dynamic BRP. We highlighted the prediction horizon T, SLR β, and the re-optimization interval Δt as the main control parameters of the algorithm, which we believe to have a major impact on its performance. Let us denote these parameters by $\mathbf{x} = [T, \beta, \Delta t]^{\mathrm{T}}$, and the metric of their goodness simply as $y(\mathbf{x}) = \sum_{i \in \mathcal{S}} t_i^{\mathrm{ef}}$. Finding the parameters $\mathbf{x}$ that minimize $y(\mathbf{x})$ in a certain operating environment is hindered by the high computing cost of the evaluating $y(\mathbf{x})$.

Surrogate-based optimization is a method that is well-suited for optimization problems with expensive function evaluations. A pseudo-code for a parallel surrogate-based optimization with the total budget of N sampling points is shown in Algorithm 1. The process is started by distributing n_{doe} points in the search space by a design of experiment method. We use the Latin hypercube sampling[18] as the design of experiment method. The method distributes the points in the search space, such that their projections onto the variable axes have a uniform spread. The objective function value y is then evaluated at the points $\{\mathbf{x}_1,\ldots,\mathbf{x}_{n_{\mathrm{doe}}}\}$.

ALGORITHM 1 A pseudo-code for parallel surrogate-based optimization

```
Distribute $n_{\mathrm{doe}}$ sampling points in the search domain using a design of experiment method.
Evaluate $y$ in parallel at points $\{\mathbf{x}_1,\ldots,\mathbf{x}_{n_{\mathrm{doe}}}\}$, obtained from the previous step.
Set $n \leftarrow n_{\mathrm{doe}} + 1$.
while $n \le N$ do
    Update the surrogate model $\hat{y}$ of $y$ based on all obtained data.
    Determine the next $n_{\mathrm{batch}}$ sampling points using the specified infill criterion.
    Evaluate $y$ in parallel at points $\{\mathbf{x}_n,\ldots,\mathbf{x}_{n+n_{\mathrm{batch}}}\}$.
    Set $n \leftarrow n + n_{\mathrm{batch}}$.
end while
return solution $\mathbf{x}$ with the smallest $y$.
```

The remaining $N - n_{\mathrm{doe}}$ sampling points are used in an iterative process, where (i) the surrogate model $\hat{y}$ representing y is updated with all data obtained so far, (ii) new sampling points are chosen based on a specified infill criterion (a procedure that involves several evaluations of the surrogate), and (iii) the objective function y is evaluated at these points. In the following, we describe the chosen surrogate model that is based on Kriging (Section 3.1) and the infill criteria (Section 3.2) in more detail.

3.1 | Kriging

Before evaluating the objective function y at some point $\mathbf{x}$, we have uncertainty of the value we are going to obtain. In Kriging (also referred to as Gaussian process regression), the observations of the function are modeled as if they are from a stochastic process, represented by the random vector

$$\mathbf{Y} = \begin{bmatrix} Y(\mathbf{x}_1), \\ \vdots \\ Y(\mathbf{x}_n) \end{bmatrix}, \tag{25}$$

having the mean of $\mathbf{1}\mu$, where $\mathbf{1}$ is a $n \times 1$ vector of ones. In this study, we model the correlation of the objective function values at points $\mathbf{x}_i$ and $\mathbf{x}_j$ using the Gaussian correlation function, defined as

$$\mathrm{Corr}\left[Y(\mathbf{x}_i), Y(\mathbf{x}_j)\right] = \exp\left(-\sum_{l=1}^{d} \theta_l \left(x_{il} - x_{jl}\right)^2\right), \tag{26}$$

where d is the number of dimensions in the search space and $\{\theta_1 \ldots \theta_d\}$ are hyperparameters that control how quickly $\mathbf{Y}$ changes in the different dimensions of the search space.

Our evaluations of objective function $y(\mathbf{x}) = \sum_{i \in \mathcal{S}} t_i^{\mathrm{ef}}$ presumably contains noise. Therefore, we define the random vector $\mathbf{Y}$ to have a covariance of

$$\mathrm{Cov}(\mathbf{Y}) = \sigma^2 \mathbf{R}_\lambda = \sigma^2 (\mathbf{R} + \lambda \mathbf{I}), \tag{27}$$

where σ^2 is variance, λ is the regularization hyperparameter (accounting for the noise), $\mathbf{I}$ is an $n \times n$ identity matrix, and $\mathbf{R}$ is the correlation matrix, defined as

$$\mathbf{R} = \begin{pmatrix} \mathrm{Corr}[Y(\mathbf{x}_1), Y(\mathbf{x}_1)] & \ldots & \mathrm{Corr}[Y(\mathbf{x}_1), Y(\mathbf{x}_n)] \\ \vdots & \ddots & \vdots \\ \mathrm{Corr}[Y(\mathbf{x}_n), Y(\mathbf{x}_1)] & \ldots & \mathrm{Corr}[Y(\mathbf{x}_n), Y(\mathbf{x}_n)] \end{pmatrix}. \tag{28}$$

$\mathbf{R}_\lambda$ is referred to as the regularized correlation matrix.

The distribution of $\mathbf{Y}$ is dependent on the hyperparameters $\mu, \theta_l, l \in \{1,\ldots,d\}, \sigma^2$, and λ. After seeing some data, these hyperparameters can be fitted by the maximum likelihood estimate.

Let us denote the sampled points after n objective function evaluations by $\mathbf{X} = [\mathbf{x}_1,\ldots,\mathbf{x}_n]^{\mathrm{T}}$ and the corresponding objective function values by $\mathbf{y} = [y_1,\ldots,y_n]^{\mathrm{T}}$. The log-likelihood function is

$$l(\mathbf{y}|\boldsymbol{\theta},\mu^2,\sigma,\lambda)= -\frac{1}{2}\left[n\ln(2\pi)+n\ln\left(\sigma^2\right)+\ln(|\mathbf{R}_\lambda|)+\frac{(\mathbf{y}-\mathbf{1}\mu)^{\mathrm{T}}\mathbf{R}_\lambda^{-1}(\mathbf{y}-\mathbf{1}\mu)}{\sigma^2}\right]. \tag{29}$$

Expressions for the optimal values of μ and σ^2 can be obtained by equating the partial derivatives $\partial l/\partial\mu$ and $\partial l/\partial\sigma^2$ to zero. The resulting expressions are

$$\begin{cases}\widehat{\mu}=\dfrac{\mathbf{1}^{\mathrm{T}}\mathbf{R}_\lambda^{-1}\mathbf{y}}{\mathbf{1}^{\mathrm{T}}\mathbf{R}_\lambda^{-1}\mathbf{1}}\\ \widehat{\sigma}^2=\dfrac{(\mathbf{y}-\mathbf{1}\mu)^{\mathrm{T}}\mathbf{R}_\lambda^{-1}(\mathbf{y}-\mathbf{1}\mu)}{n}\end{cases} \tag{30}$$

Substituting Equation (30) to (29), and omitting the constant terms, yields the so-called "concentrated log-likelihood"

$$-\frac{n}{2}\ln\left(\widehat{\sigma}^2\right)-\frac{1}{2}\ln(|\mathbf{R}_\lambda|), \tag{31}$$

This form only requires finding the optimal values for $\theta_l, l\in\{1,...,d\}$ and λ, after which $\widehat{\mu}$ and $\widehat{\sigma}^2$ can be obtained by substitutions.

After choosing the hyperparameters, the model can be used to make a prediction of the objective function value y at point $\mathbf{x}^*$ that has not been evaluated. Let us define a new correlation vector $\mathbf{r}$ as

$$\mathbf{r}=\begin{bmatrix}\mathrm{Corr}[Y(\mathbf{x}^*),Y(\mathbf{x}_1)]\\ \vdots\\ \mathrm{Corr}[Y(\mathbf{x}^*),Y(\mathbf{x}_n)]\end{bmatrix}. \tag{32}$$

The predicted mean $\widehat{y}(\mathbf{x}^*)$ and variance $\widehat{s}^2(\mathbf{x}^*)$ are then (see the derivation in 59)

$$\begin{cases}\widehat{y}(\mathbf{x}^*)=\widehat{\mu}+\mathbf{r}^{\mathrm{T}}\mathbf{R}_\lambda^{-1}(\mathbf{y}-\mathbf{1}\mu)\\ \widehat{s}^2(\mathbf{x}^*)=\widehat{\sigma}^2\left[1+\lambda-\mathbf{r}^{\mathrm{T}}\mathbf{R}_\lambda^{-1}\mathbf{r}\right].\end{cases} \tag{33}$$

3.2 | Infill criteria

Finding the optimum of y based on the surrogate $\widehat{y}$ requires balancing between sampling points that $\widehat{y}$ indicates to be competitive (i.e., exploitation) and sampling points that lead to improvements in the accuracy of $\widehat{y}$ (i.e., exploration). $\widehat{y}$ should be accurate especially in regions where competitive points are found. The new points are chosen based on an infill criterion. A natural infill criterion for exploiting $\widehat{y}$ is to sample the point $\mathbf{x}^*$ that minimizes $\hat{y}(\mathbf{x}^*)$ in Equation (33). This is referred to as the prediction-based infill criterion.

A commonly used infill criterion in surrogate-based optimization is to sample the point $\mathbf{x}^*$ that maximizes the expected improvement *E(I)* in the best known objective function value. This infill criterion has a balanced exploration and exploitation. The expected improvement is defined as

$$E(I)=\int_{I=0}^{I=\infty} I\left\{\frac{1}{\sqrt{2\pi}s(\mathbf{x})}\exp\left[-\frac{(y_{\min}-I-\widehat{y}(\mathbf{x}))^2}{2s^2(\mathbf{x})}\right]\right\}dI, \tag{34}$$

where $y_{\min}$ is the best known objective function value.[25] Using integration by parts, the equation can be transformed into

$$E(I)=s(\mathbf{x})[u\Phi(u)+\phi(u)] \tag{35}$$

where $\Phi(\cdot)$ and $\phi(\cdot)$ are the cumulative distribution function and the density function of the normal distribution, respectively, and

$$u=\frac{f_{\min}-\widehat{y}(\mathbf{x})}{s(\mathbf{x})}. \tag{36}$$

In this study, we sample the first n_{ei} infill points using the expected improvement criterion (maximization of Equation 35) and finalize the search by sampling $n_{\widehat{y}}$ infill points using the prediction-based criterion (minimization of $\hat{y}(\mathbf{x}^*)$ in Equation 33). In order to reduce the total time of an optimization procedure we use a batch size of $n_{\mathrm{batch}}=3$ when sampling using the expected improvement criterion. The parallel infill points are sampled using the method by Ginsbourger et al.[33] The final infill points based on prediction-based criterion are evaluated in series ($n_{\mathrm{batch}}=1$). The method we use follows fairly closely the original EGO method[25] with the exceptions that we use a slightly different correlation function (Equation 26) and finalize the search by the prediction-based infill criterion. Despite of these differences, let us refer to the described method as EGO for the sake of brevity. As the implementation of the method, we use EGO function from the Python module SMT.[60] A reader interested in further information on surrogate-based optimization may consult the works by Jones[61] and Forrester et al.[62]

4 | RESULTS

We formulate real-time optimization problems based on data recorded from the bike sharing system in the Helsinki region. The origin–destination data of the trips conducted in the system, as well as the locations and capacities of its stations, have been published by Helsinki Region Transport[55,63] since 2016 under the CC BY 4.0 license. Kainu[64] has published the number of available bikes at the stations of the system at the sampling interval of 5 min since 2017 (also under CC BY 4.0). In 2020, the system consisted of 352 stations. We have chosen a subset of these stations that lay inside the rectangular region, specified by the coordinate ranges [60.15 … 60.19°N] and [24.90 … 25.00°E] (Figure 5). The region, capturing the city center of Helsinki, has a total of 92 stations.

We use the origin–destination data recorder in June 2020 as the training data, and the corresponding data recorded during August 17–28, 2020, as the test data**. For the sake of simplicity, we only consider weekdays in this work, as the demand profiles are different on weekdays and weekends, and should anyways be handled separately. The number of weekdays in June 2020 was 22, whereas the specified

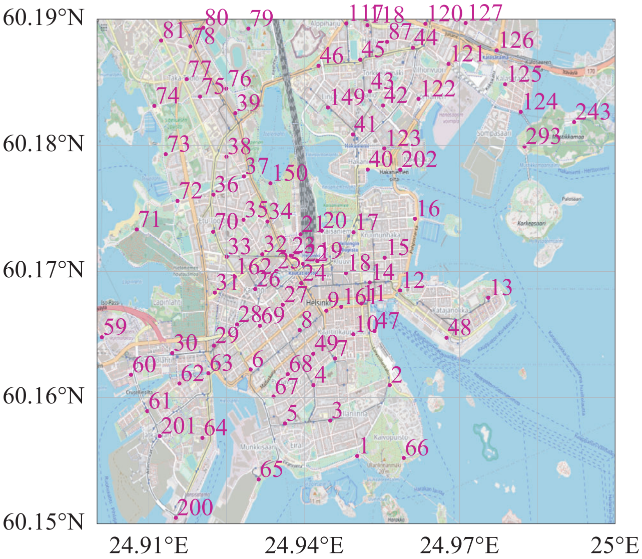


FIGURE 5 The locations of 92 bike stations around the city center of Helsinki that are used in this study. The map is generated using OpenStreetMap[66]

time window in August 2020 includes 10 weekdays. Here, training the prediction model means determining the pickup and return rates, $\mu(t)_i$ and $\lambda(t)_i, i \in \mathcal{S}$, from the aforementioned origin–destination data. We determine the rates by calculating the mean of pickups and returns during 15-min intervals; the resulting values are maximum likelihood estimates for Poisson variables. Figure 1 shows the obtained rates for the station on Unioninkatu (i.e., station 11 in Figure 5).

When evaluating an online algorithm on the test data, we initiate the system at 6:00 using the numbers of bikes at the stations from the data recorded by Kainu.[64] We then repeatedly perform re-optimization procedures, and simulate the evolution of the system (Figure 2). At $t_{\text{end}} = 15{:}00$, the rebalancing is stopped. Finally, we simulate the system until 22:00 (without rebalancing) and determine the total time, $\sum_{i \in \mathcal{S}} t_i^{\text{ef}}$, the stations in the system were either empty or full during the time window of 6:00–22:00. The goodness of an online algorithm is measured based on $\sum_{i \in \mathcal{S}} t_i^{\text{ef}}$, averaged over the 10 test days.

Unlike some other bike sharing systems, the system in Helsinki allows bikes to be returned to full stations. However, in order to keep the methods analogous to other systems, we preclude returning a bike to a full station in our simulations, and use the number of docks C_i as the initial number of bikes if the station was initially overfull. We also wish to point out that in the origin–destination data, the pickup demand is censored (i.e., the demand is not visible in the data) during the time periods when the station in question is empty. The return demand, on the other hand, is not censored in the data when the station is full, as the system allows stations to be overfull. We assume the censored pickup demand to have only a small effect on our evaluations.

Regarding the model parameters, we determine the elements of the distance matrix $d_{i,j}$ as the Euclidean distance between the stations with a detour factor of 1.4. The vehicle(s) have a capacity of $Q_v = 22$. Picking up or delivering a bike from/to a station takes 1 min. These three parameters are the same as in the work by Schuijbroek et al.[50] Further, we define vehicles to have an average speed of $\overline{v} = 20$ km/h. Thus, based on the average speed and the pickup/delivery time, parameters d^- and d^+ have the value of 333 m. In the beginning of operation, the load on the vehicle(s) is half of their capacity, $q_v^{\text{init}} = 11, v \in \mathcal{V}$.

In the following, we present the results from test cases with one (Sections 4.1 and 4.2) and two rebalancing vehicles (Section 4.3). In all cases, we allocate 32 evaluations for the EGO method, described in Section 3, and use the following sampling plan: first, we evaluate $n_{\text{doe}} = 15$ initial points, then $n_{\text{EI}} = 15$ infill points using the expected improvement criterion in batches of $n_{\text{batch}} = 3$, and finally $n_{\hat{y}} = 2$ infill points in series using the prediction-based criterion. We also show reference results generated using full factorial sampling. In this method, the variables, defining the search space, are discretized evenly, and all variable combinations are evaluated. In order to ensure that the full factorial sampling is sufficiently dense, we allocate 64 evaluations for it.

The evaluations were performed on a high performance computing facility, consisting Intel Xeon E5 2680/2690 v3 central processing units (CPUs). The number of allocated threads to evaluate a day in the test data was chosen to be the same as the number of rebalancing vehicles in the test case. Each evaluation was allocated 4 GB of memory. We use Gurobi 9.1.2 to solve the MIP models (i.e., the routing model [Equations 7–23] and the clustering model, defined in 50). Regarding the solver time limits, if only one vehicle is used in rebalancing, we define the solver time limit to equal the re-optimization interval[††] Δt. If multiple vehicles are used, we define the time limit to solve the clustering model as 20% of Δt, and the time limit to solve the resulting routing models as $\Delta t - t_{\text{clu}}$, where t_{clu} is the run time of solving the clustering model (the routing models are solved in parallel).

4.1 | Two-dimensional case with one vehicle

Let us start with a case where one rebalancing vehicle operates in the region shown in Figure 5 (the vehicle has the initial location of $l_1^{\text{init}} = 1$). In this case, the purpose is to optimize the SLR β and the prediction horizon T while keeping the re-optimization interval fixed at $\Delta t = 1$ h. The value represents a natural choice for the re-optimization interval that could be chosen in practice. We specify the search domain using the bounds $\beta \in [0.1, \ldots, 0.95]$ and $T \in [0.1, \ldots, 6.0]$ h.

Figure 6 illustrates the results obtained by the EGO method and the full factorial sampling. The dots in both subfigures represent the initial sampling points. In Figure 6A, the crosses indicate the locations of the iteratively evaluated infill points. After the optimization procedure with EGO method, we trained a surrogate using all observed data. The contour in the Figure 6A represents the predicted objective function value ($\hat{y}(\boldsymbol{x}^*)$) in Equation 33) in the search space based on this surrogate.

Both figures indicate the design space to have a vertically aligned valley of good control parameter combinations. The valley starts at the top of the search space at around (β = 0.8, T = 6.0 h) and becomes wider when Δt is reduced. The valley is surrounded by two separate regions with poorly performing control parameter combinations. The wide region at the top left corner of the search space corresponds to algorithms where the SLR β is so low that no or only a few stations are added to the route of the vehicle. This leads to situations where the vehicle is on idle and waits for instructions. If no rebalancing is performed during the test days, the objective function has the value of $\sum_{i\in\mathcal{S}} t_i^{\text{ef}} = 245.49$ h (applies to several points in this region). On the other hand, the (vertically) narrow region at the top right corner of the search space corresponds to algorithms where too many stations are added to the routing model, leading to combinatorially complex models. As a result, the MIP solver fails to find an integer solution in the given computing time, and again no rebalancing instructions can be sent to the vehicle.

The points with the smallest objective function values are circled in Figure 6. The numeric values of these points, as well as the corresponding objective function values, are listed in Table 1. With the control parameters obtained by the EGO method the total time the stations are empty or full, $\sum_{i\in\mathcal{S}} t_i^{\text{ef}}$, is 3.32% smaller than with those obtained by the full factorial sampling.

Figure 7 shows the realized and planned route of the vehicle in the morning on August 17, 2020 when using the optimized control parameters obtained by the EGO method (see the map in Figure 5). The realized and planned routes are indicated by solid and dashed lines, respectively. The three subfigures correspond the first three time points of the day when updates of the optimized operations are sent to the vehicle. Figure 8 shows the realized and planned load on the vehicle corresponding to these states. These two figures illustrate how the optimized plan of operations changes when new information is obtained. In Figure 8C, the one-bike gap between the realized and the planned load is caused by a shortage of bikes at station 121, precluding the vehicle to follow the planned operations precisely.

With the optimized control parameters, obtained by the EGO method, the average solution time of the MIP solver was 130.9 s, indicating that in many cases the solver did not need all of the allocated solution time of $\Delta t = 3600$ s to find the optimal operations. However, the solution times had significant variance. The minimum and maximum solution times were 0.0015 and 3600.0 s, respectively. The MIP gap varied from 0.00% to 2.33%.

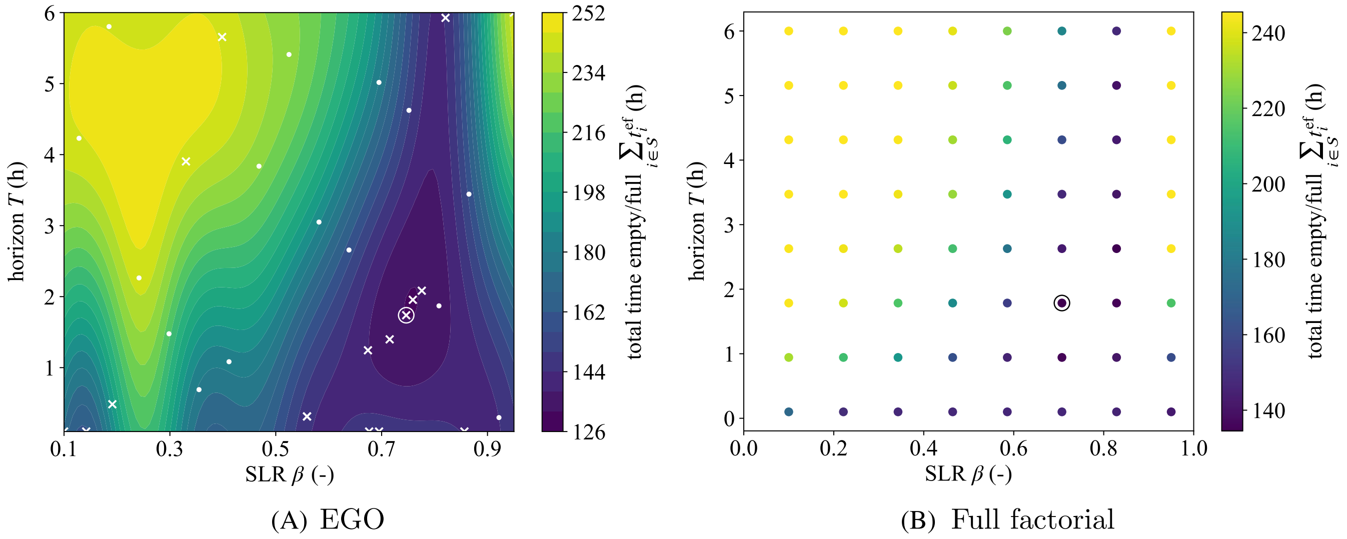
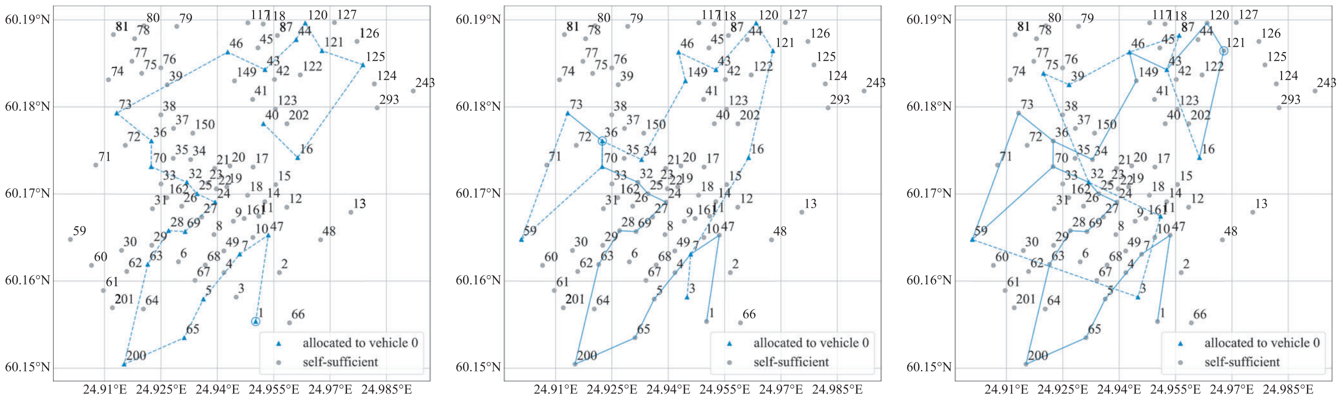


FIGURE 6 Results obtained using the EGO method (A) and full factorial sampling (B). The dots and crosses represent the initial and infill sampling point, respectively. The best obtained point is circled. The contour in (A) is from the surrogate that is trained with all observed data. EGO, efficient global optimization; SLR, service level requirement

TABLE 1 Optimized SLR β and horizon T, and the corresponding objective function values

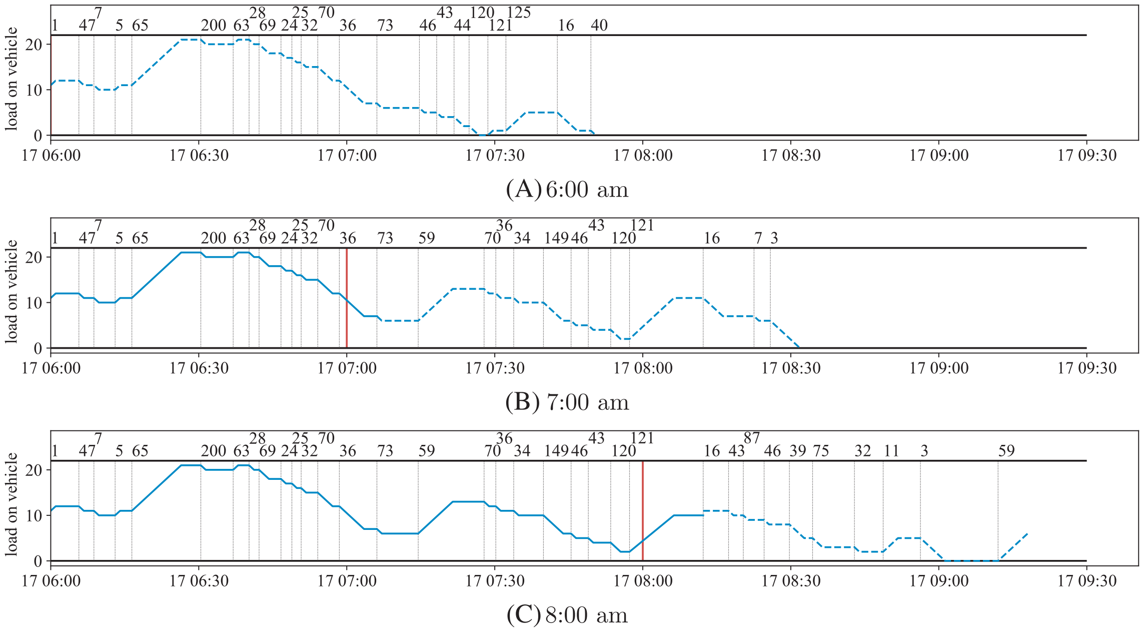
Method	Objective $\sum_{i\in\mathcal{S}} t_i^{\text{ef}}$ (h)	Parameters x SLR β (−)	 Horizon T (h)	Evaluations
EGO	130.05	0.746	1.738	32
Full factorial	134.52	0.707	1.786	64

Note: The re-optimization interval has the fixed value of $\Delta t = 1$ h.
Abbreviations: EGO, efficient global optimization; SLR, service level requirement.



(A) 6:00 am (B) 7:00 am (C) 8:00 am

FIGURE 7 Realized (solid line) and planned route (dashed line) of the vehicle when obtaining first three operations updates at 6:00 am (A), 7:00 am (B), and 8:00 am (C) on August 17, 2020



(A) 6:00 am

(B) 7:00 am

(C) 8:00 am

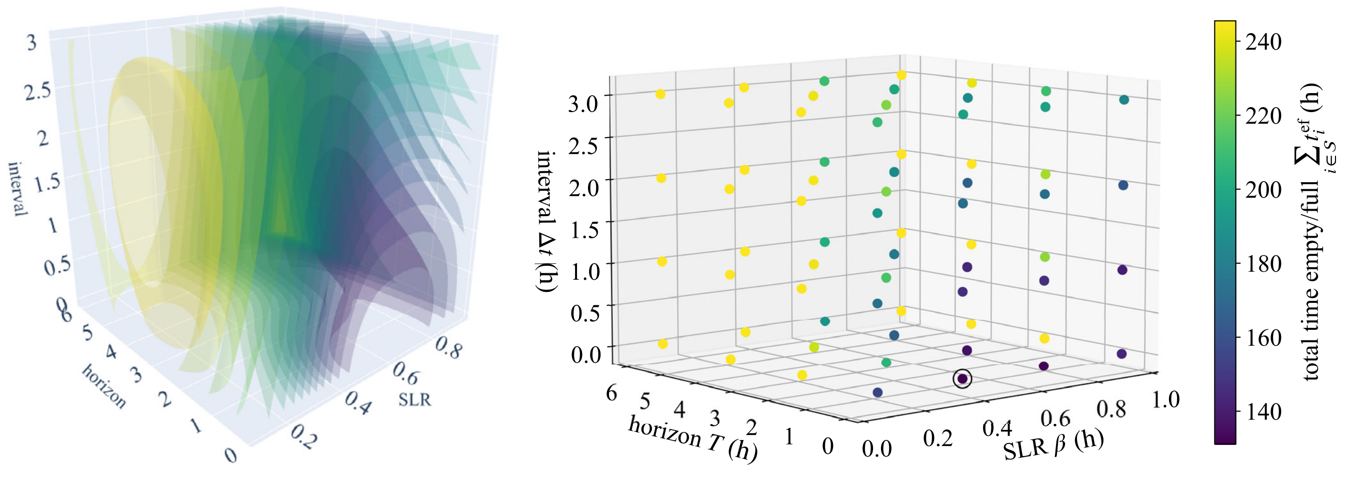
FIGURE 8 Realized (solid line) and planned load (dashed line) on the vehicle corresponding to the three states shown in Figure 7. The vertical dotted lines indicate the arrival times to specific stations

4.2 | Three-dimensional case with one vehicle

Considering that the MIP solver did not need most of the allocated solution time in the case, the next question is whether we could find a better algorithm if we also relax the re-optimization interval Δt and include it as a new dimension in the search space. In this section, we study the same dynamic BRP as in the previous section but seek the optimal control parameter combination from the three-dimensional search space, defined by bounds $\beta \in [0.1\ldots0.95]$, $T \in [0.1\ldots6.0]$ h, and $\Delta t \in [0.01\ldots3.0]$ h.

We use the 64 sampling points allocated to the full factorial sampling by discretizing each dimension using four points ($4^3 = 64$). Figure 9B illustrates the results obtained at these sampling points. It is worth noticing that the objective function landscape at all four 4 × 4 horizontal layers resemble that of Figure 6B. Figure 9A shows the isosurfaces obtained by a surrogate that is trained with all observed data. Table 2 lists the best control parameters, and the objective function values, obtained by both methods. Despite having 50% fewer function evaluations, the EGO method yielded a control parameter combination with 1.15% smaller objective function value than the full factorial sampling.

The improvement in the best obtained objective function value is only 0.30% in comparison to the previous two-dimensional case (compare Tables 1 and 2) although the re-optimization procedures are performed more frequently. Arguably, there could be at least two reasons for observing only a relatively small improvement. First, we have



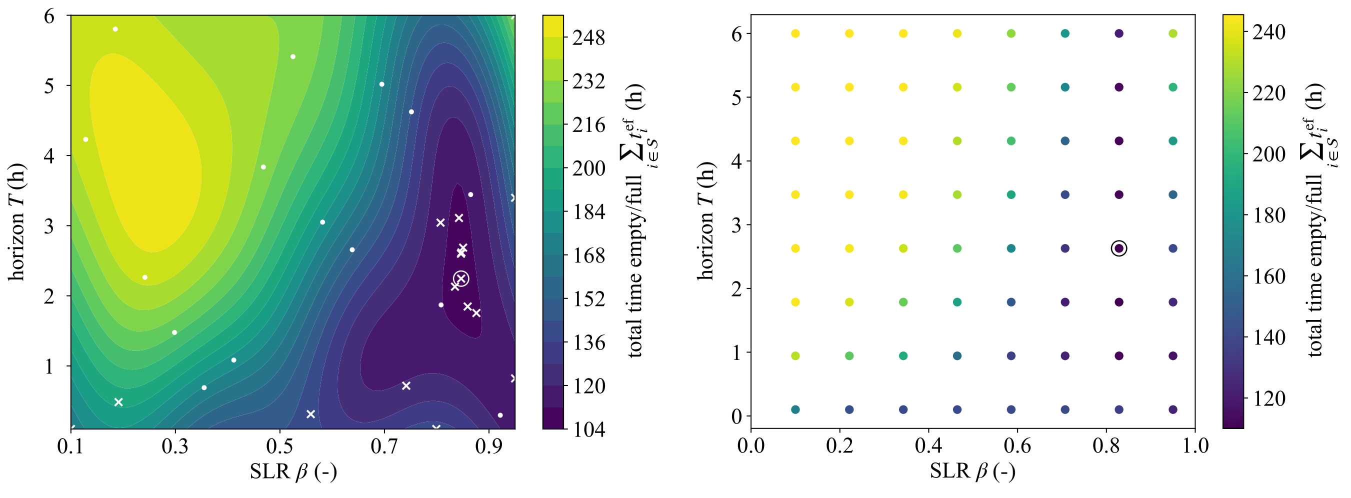
(A) EGO (B) Full factorial

FIGURE 9 Results obtained using the EGO method (A) and full factorial sampling (B). The isosurfaces in (A) are from the surrogate that is trained with all observed data. The best control parameter combination obtained by full factorial sampling is circled. EGO, efficient global optimization; SLR, service level requirement

TABLE 2 Optimized SLR β, horizon T, and re-optimization interval Δt, as well as the corresponding objective function values

	Objective	Parameters x			
Method	$\sum_{i \in S} t_i^{ef}$ (h)	SLR β (−)	Horizon T (h)	Interval Δt (h)	Evaluations
EGO	129.66	0.569	0.445	0.142	32
Full factorial	131.18	0.383	0.100	0.010	64

Abbreviations: EGO, efficient global optimization; SLR, service level requirement.



(A) EGO (B) Full factorial

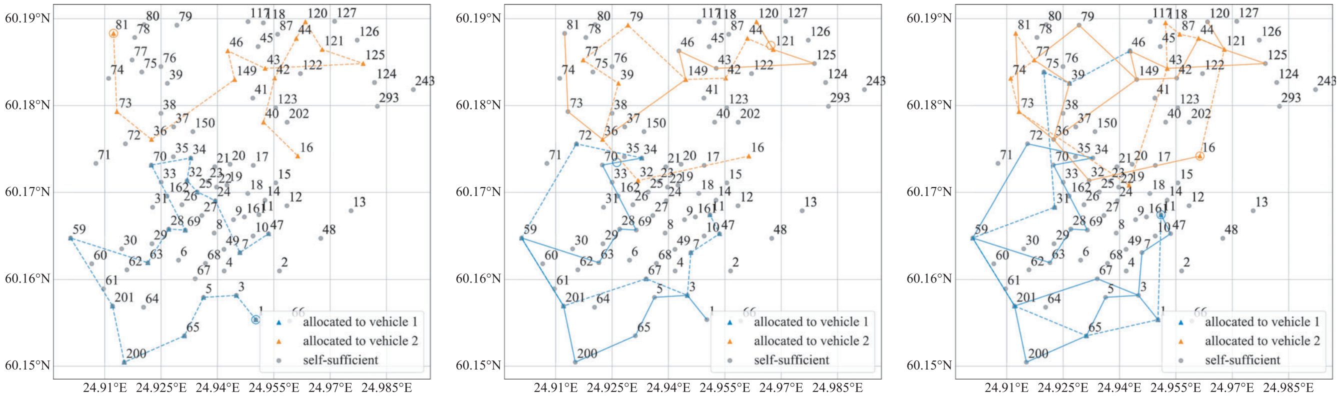
FIGURE 10 Results obtained using the EGO method (A) and full factorial sampling (B) on the two-dimensional cse with two vehicles. The dots and crosses represent the initial and infill sampling point, respectively. The best obtained points are circled. EGO, efficient global optimization; SLR, service level requirement

allocated the same number of evaluations for the methods in the two cases, despite the search space of the latter having an additional dimension. Therefore, the search space in the first case can be examined more thoroughly than that of the second case. Second, the performance of the online algorithm seems to have only a small dependency on the re-optimization interval Δt. This argument is supported by the surrogate that is trained with all 32 evaluated data points by the EGO method. Especially in the bottom region of the

TABLE 3 Optimized control parameters, and the corresponding objective function values, on the two-dimensional case with two vehicles

	Objective	Parameters x		
Method	$\sum_{i \in \mathcal{S}} t_i^{\text{ef}}$ (h)	SLR β (−)	Horizon T (h)	Evaluations
EGO	109.21	0.847	2.245	32
Full factorial	110.05	0.829	2.629	64

Note: The re-optimization interval has the fixed value of $\Delta t = 1$ h.
Abbreviations: EGO, efficient global optimization; SLR, service level requirement.



(A) 6:00 am (B) 7:00 am (C) 8:00 am

FIGURE 11 Realized (solid line) and planned route (dashed line) of the two vehicles when obtaining first three operations updates at 6:00 am (A), 7:00 am (B), and 8:00 am (C) on August 17, 2020

cube (where Δt is small), the isocurves are aligned nearly vertically. This indicates low dependency of the re-optimization interval Δt. Further, the hyperparameters $\{\theta_1,\ldots,\theta_d\}$ of the Kriging model give an indication of the importance of the different variables. The hyperparameters θ_1 and θ_2, corresponding to β and T have values of 0.708 and 0.454, respectively, whereas θ_3, corresponding to Δt has a value of 0.101, indicating lesser importance.

4.3 | Two-dimensional case with two vehicles

Finally, let us examine a case with two rebalancing vehicles. As the inclusion of the re-optimization interval Δt in the search space provided only a minor improvement in the objective function value, we fix it, for the sake of simplicity, again to the value of $\Delta t = 1$ h. Thus, the case is otherwise the same as that of Section 4.1, but has two rebalancing vehicles instead of one. The vehicles have initial locations of $l_1^{\text{init}} = 1$ and $l_2^{\text{init}} = 81$, located at the opposite sides of the region.

Figure 10 illustrates the results with the EGO method and full factorial sampling. The objective function landscape is similar to that of Figure 6 but, in this case, the valley of good control parameter combinations has shifted slightly toward top right. The shift is due to the clustering method, which reduces the combinatorial complexity of the routing models. Even at the upper bound of the SLR $\beta = 0.95$, the MIP solver can find an integer solution in most of the optimization procedures. The control parameters with the best objective function values have also shifted slightly toward the top right corner of the search space (Table 3). In this case, the EGO method yielded control parameters with an objective function value 0.76% lower than that from the full factorial sampling.

Figure 11 illustrates the routes of the vehicles when they receive the first three operations updates in the morning on August 17, 2020. Here, the re-optimization of operations may reallocate a station from one vehicle to another (see, e.g., station 32 in the optimized operations at time points 6:00 and 7:00).

We tested also an alternative infill strategy for the EGO method where also the first n_{ei} infill points are generated based on the prediction-based infill criterion. The optimized objective function values in the three test cases (Sections 4.1–4.3) were within 0.7% from those of the primary strategy described earlier in this section. Arguably, if the objective function has multiple local minima, the alternative strategy may yield sub-optimal solutions, as it is essentially a local search method.

5 | DISCUSSION AND CONCLUSIONS

In this study, we have investigated surrogate-based optimization of a periodic rescheduling algorithm on a dynamic BRP. We optimized three continuous control parameters of the algorithm (the SLR β, prediction horizon T and re-optimization interval Δt) and investigated how these parameters affect the performance of the algorithm.

Based on our results on two- and three-dimensional search spaces, the main benefit of using the surrogate-based optimization is the reduction in the computational cost of tuning an online algorithm to a certain environment. The full factorial sampling was allocated 64 evaluations, all of which involve evaluating the performance of the algorithm on 10 days, each having 9 h of rebalancing vehicle operation. Thus, the total allocated computing time to solve the MIP models was 5760 CPU-hours. On the three studied cases, the surrogate-based optimization method was able to find control parameter combinations with 0.76% to 3.32% better objective function values with 50% fewer function evaluations in comparison to the full factorial sampling. However, the actual computing times of evaluating a point varied significantly depending on the region of the search space. In general, points with the smallest SLR β had the smallest computing times (in an order of seconds) due to the low combinatorial complexity of the resulting MIP models, whereas the points with a largest SLR β used, in many cases, all of the allocated computing time.

We can identify a few directions for improving the method. First, while the Latin hypercube sampling is a frequently used design of experiment method, yielding a uniform projected spread along the variable dimensions, it is not necessarily a space-filling method. Such a design of experiment method is, for example, that by Morris and Mitchell,[19] which maximizes the minimum distance among the points of a Latin hypercube. Second, cost-aware methods have recently been developed at least in the field of Bayesian optimization.[65] Considering the high variance that we observed in the actual computing cost, these methods could be useful in reducing the computing time of the optimization procedures, as they construct also a surrogate for the evaluation time and use this surrogate when deciding new sampling points. Third, surrogate-based optimization is applicable to optimization problems with up to around 20 variables[‡‡].[20,35] The future study should investigate expanding the dimensions of the search space by other continuous parameters related to an online scheduling algorithm (e.g., the allocated computing time per re-optimization procedure). A pre-screening of parameters by a fractional factorial design[17] could be useful prior to surrogate-based optimization, in order to include only important parameters in the optimization.

On the second test case, in which we also optimized the re-optimization interval, the best obtained control parameter combination has the re-optimization interval of $\Delta t = 0.142$ h (i.e., around 8.5 minutes). Sending updated operations to the drivers of the vehicles this frequently would be very impractical in reality, as the drivers would have uncertainty of the operations they need to perform after visiting the next planned station. Therefore, most operators would prefer to use the re-optimization interval of $\Delta t = 1$ h of the first case, the objective function value of which is only marginally compromised. This is also the case in most industries facing varying demand patterns or disturbances in the processes. Once an operator decides to take an action, for example, perform a small maintenance routine, he is occupied for a certain time and this routine may also limit or determine the next production decisions. For instance, in multi-product processes, a sequence-dependent changeover may take even 30 minutes (in, e.g., Pulp & Paper or steel industry) and there is no rationale to swap the next product during this time.

As already indicated above, the methodology described in this paper is generic and can be applied to various processes. The dynamics of a bike sharing system and process industries are both stochastic by nature. On the material side, there may be urgent orders or supply problems from the upstream processes. Additionally, the processing times are heavily affected by potential disturbances or deviations in, for example, material qualities and yields. Many of these stochastic elements cannot be fully prepared for but may be observed or predicted, for example, by surrogate models, allowing to better tune the optimization model parameters, as can also be seen in many earlier contributions in PSE, cited in the literature overview. The main challenge is to have access to sufficient information and suitable methodologies in order to better describe the underlying process model through historical data. The discussed study, thus, also contributes to the research on the integration of scheduling and control: here, we use AI/ML methods to acquire more information and knowledge about the low-level process dynamics (that in this case cannot be fully observed nor controlled) in order to improve the upper-level awareness and, consequently, the overall decision quality.

ACKNOWLEDGMENTS

Financial support from the Academy of Finland is gratefully acknowledged (project RELOOP, decision number 330388, and project DeVeMaPa, decision number 336092). The calculations presented in this study were performed using computer resources within the Aalto University, School of Science, "Science-IT" project. Finally, the authors would like to acknowledge Helsinki Regional Transport (HSL) for making the bike sharing data available, and Markus Kainu for collecting the numbers of bikes at the stations as a function of time.

AUTHOR CONTRIBUTIONS

Teemu J. Ikonen: Conceptualization (equal); data curation (lead); funding acquisition (equal); investigation (equal); methodology (lead); software (lead); visualization (lead); writing – original draft (lead). **Keijo Heljanko:** Funding acquisition (equal); investigation (supporting); supervision (supporting); validation (equal); writing – review and editing (equal). **Iiro Harjunkoski:** Conceptualization (equal); funding acquisition (equal); investigation (equal); methodology (supporting); supervision (lead); validation (equal); writing – original draft (supporting); writing – review and editing (equal).

ENDNOTES

[*] This dataset can be generated from user trip data, containing the identifiers of the origin and destination stations, as well as the corresponding time stamps. Such data are widely published by the operators of bike sharing systems.

[†] The estimate is based on an assumption that two trips belong to the same batch, if they pairwise have the same pickup and return stations and their corresponding time stamps are within 3 min.

[‡] The method takes steps using the fifth-order Runge–Kutta method and controls the error by assuming an accuracy of the fourth-order method.

[§] This assumption has the caveat that the algorithm cannot prioritize a station with high demand over a station with low demand. Nevertheless, station-specific service level requirements are outside the scope of this study.

[¶] The routing model with multiple vehicles has a high combinatorial complexity already with relatively small number of stations. The authors use it mainly as a reference model.

[**] We use the data from the second half of August as the test data because July and the first half of August are the main vacation times in Finland. The vacations presumably have an effect on the daily demand profiles of the bike sharing system. Studying this effect falls outside the scope of this study.

[††] We omit the computing time to determine the bounds $s_i^{\min}$ and $s_i^{\max}, i \in \mathcal{S}$ (Section 2.3.1), as this time is typically in an order of seconds, or tens of seconds.

[‡‡] With a greater number of variables the repeatedly performed retraining the surrogate may incur a prohibitively large computing cost.

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are available from the corresponding author upon reasonable request.

ORCID

Teemu J. Ikonen https://orcid.org/0000-0001-7413-9310
Keijo Heljanko https://orcid.org/0000-0002-4547-2701
Iiro Harjunkoski https://orcid.org/0000-0003-0428-0751

REFERENCES

1. Méndez CA, Cerdá J, Grossmann IE, Harjunkoski I, Fahl M. State-of-the-art review of optimization methods for short-term scheduling of batch processes. *Comput Chem Eng*. 2006;30(6–7):913-946. doi: 10.1016/j.compchemeng.2006.02.008
2. Harjunkoski I, Maravelias CT, Bongers P, et al. Scope for industrial applications of production scheduling models and solution methods. *Comput Chem Eng*. 2014;62:161-193. doi:10.1016/j.compchemeng.2013.12.001
3. Castro PM, Grossmann IE, Zhang Q. Expanding scope and computational challenges in process scheduling. *Comput Chem Eng*. 2018;114: 14-42. doi:10.1016/j.compchemeng.2018.01.020
4. Gupta D, Maravelias CT. On the design of online production scheduling algorithms. *Comput Chem Eng*. 2019;129:106517. doi:10.1016/j.compchemeng.2019.106517
5. Sabuncuoglu I, Kizilisik OB. Reactive scheduling in a dynamic and stochastic FMS environment. *Int J Prod Res*. 2003;41(17):4211-4231. doi:10.1080/0020754031000149202
6. Gupta D, Maravelias CT, Wassick JM. From rescheduling to online scheduling. *Chem Eng Res Des*. 2016;116:83-97. doi:10.1016/j.cherd.2016.10.035
7. Gupta D, Maravelias CT. On deterministic online scheduling: major considerations, paradoxes and remedies. *Comput Chem Eng*. 2016;94: 312-330. doi:10.1016/j.compchemeng.2016.08.006
8. Stevenson Z, Fukasawa R, Ricardez-Sandoval L. Evaluating periodic rescheduling policies using a rolling horizon framework in an industrial-scale multipurpose plant. *J Sched*. 2020;23(3):397-410. doi: 10.1007/s10951-019-00627-5
9. Dong Y, Maravelias CT, Jerome NF. Reoptimization framework and policy analysis for maritime inventory routing under uncertainty. *Optim Eng*. 2018;19(4):937-976. doi:10.1007/s11081-018-9383-8
10. Touretzky CR, Harjunkoski I, Baldea M. Dynamic models and fault diagnosis-based triggers for closed-loop scheduling. *AIChE J*. 2017; 63(6):1959-1973. doi:10.1002/aic.15564
11. Katragjini K, Vallada E, Ruiz R. Flow shop rescheduling under different types of disruption. *Int J Prod Res*. 2013;51(3):780-797. doi: 10.1080/00207543.2012.666856
12. Pattison RC, Touretzky CR, Harjunkoski I, Baldea M. Moving horizon closed-loop production scheduling using dynamic process models. *AIChE J*. 2017;63(2):639-651. doi:10.1002/aic.15408
13. Ikonen TJ, Heljanko K, Harjunkoski I. Reinforcement learning of adaptive online rescheduling timing and computing time allocation. *Comput Chem Eng*. 2020;141:106994. doi:10.1016/j.compchemeng.2020.106994
14. Dalle Ave G, Alici M, Harjunkoski I, Engell S. An explicit online resource-task network scheduling formulation to avoid scheduling nervousness. In: Kiss AA, Zondervan E, Lakerveld R, Özkan L, eds. *Computer Aided Chemical Engineering*. Vol 46. Elsevier; 2019:61-66. doi:10.1016/B978-0-12-818634-3.50011-4
15. Dong Y, Maravelias CT. Terminal inventory level constraints for online production scheduling. *Eur J Oper Res*. 2021;295:102-117. doi: 10.1016/j.ejor.2021.02.029
16. Vieira GE, Herrmann JW, Lin E. Rescheduling manufacturing systems: a framework of strategies, policies, and methods. *J Sched*. 2003;6(1): 39-62. doi:10.1023/A:1022235519958
17. Montgomery DC. *Design and Analysis of Experiments*. John Wiley & Sons; 2017. doi:10.1007/978-3-319-52250-0
18. McKaya MD, Beckmana RJ, Conover WJ. Comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Dent Tech*. 1979;21(2):239-245. doi: 10.2307/1268522
19. Morris MD, Mitchell TJ. Exploratory designs for computational experiments. *J Stat Plan Inference*. 1995;43(3):381-402. doi:10.1016/0378-3758(94)00035-T
20. Frazier PI. A tutorial on Bayesian optimization. *ArXiv*. [Preprint] arXiv: 180702811, 2018.
21. Kushner HJ. A new method of locating the maximum point of an arbitrary multipeak curve in the presence of noise. *J Basic Eng*. 1964; 86(1):97-106. doi:10.1115/1.3653121
22. Močkus J. On Bayesian methods for seeking the extremum. *Optimization Techniques IFIP Technical Conference*. Springer; 1975:400-404. doi:10.1007/3-540-07165-2_55
23. Močkus J, Tiesis V, Žilinskas A. The application of Bayesian methods for seeking the extremum. In: Dixon LCW, Szego GP, eds. *Towards Global Optimisation*. Vol 2. North-Holand; 1978:117-129.
24. Sacks J, Welch WJ, Mitchell TJ, Wynn HP. Design and analysis of computer experiments. *Stat Sci*. 1989;4:409-423.
25. Jones DR, Schonlau M, Welch WJ. Efficient global optimization of expensive black-box functions. *J Glob Optim*. 1998;13(4):455-492. doi:10.1023/A:1008306431147
26. Broomhead DS and Lowe D. Radial Basis Functions, Multi-Variable Functional Interpolation and Adaptive Networks. Technical report, Royal Signals and Radar Establishment Malvern (United Kingdom), 1988.
27. Krige DG. A statistical approach to some basic mine valuation problems on the witwatersrand. *J South Afr Inst Min Metall*. 1951;52(6): 119-139.
28. Frazier P, Powell W, Dayanik S. The knowledge-gradient policy for correlated normal beliefs. *INFORMS J Comput*. 2009;21(4):599-613. doi:10.1287/ijoc.1080.0314
29. Srinivas N, Krause A, Kakade SM, Seeger M. Gaussian process optimization in the bandit setting: no regret and experimental design. *ArXiv*. [Preprint] arXiv:09123995. 2009. doi:10.1109/TIT.2011.2182033
30. Keane AJ. Statistical improvement criteria for use in multiobjective design optimization. *AIAA J*. 2006;44(4):879-891. doi:10.2514/1.16875
31. Knowles J. ParEGO: A hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems.

IEEE Trans Evol Comput. 2006;10(1):50-66. doi:10.1109/TEVC.2005.851274

32. Forrester AIJ, Sóbester A, Keane AJ. Multi-fidelity optimization via surrogate modelling. *Proc R Soc A Math Phys Eng Sci*. 2007;463(2088):3251-3269. doi:10.1098/rspa.2007.1900
33. Ginsbourger D, Le Riche R, Carraro L. Kriging is well-suited to parallelize optimization. *Computational Intelligence in Expensive Optimization Problems*. Springer; 2010:131-162. doi:10.1007/978-3-642-10701-6_6
34. Haftka RT, Villanueva D, Chaudhuri A. Parallel surrogate-assisted global optimization with expensive functions–a survey. *Struct Multidiscip Optim*. 2016;54(1):3-13. doi:10.1007/s00158-016-1432-3
35. Forrester AIJ, Keane AJ. Recent advances in surrogate-based optimization. *Prog Aerosp Sci*. 2009;45(1–3):50-79. doi:10.1016/j.paerosci.2008.11.001
36. Greenhill S, Rana S, Gupta S, Vellanki P, Venkatesh S. Bayesian optimization for adaptive experimental design: a review. *IEEE Access*. 2020;8:13937-13948. doi:10.1109/ACCESS.2020.2966228
37. McBride K, Sundmacher K. Overview of surrogate modeling in chemical process engineering. *Chem Ing Tech*. 2019;91(3):228-239. doi:10.1002/cite.201800091
38. Wan X, Pekny JF, Reklaitis GV. Simulation-based optimization with surrogate models—application to supply chain management. *Comput Chem Eng*. 2005;29(6):1317-1328. doi:10.1016/j.compchemeng.2005.02.018
39. Sahay N, Ierapetritou M. Multienterprise supply chain: simulation and optimization. *AIChE J*. 2016;62(9):3392-3403. doi:10.1002/aic.15399
40. Shi H, You F. A novel adaptive surrogate modeling-based algorithm for simultaneous optimization of sequential batch process scheduling and dynamic operations. *AIChE J*. 2015;61(12):4191-4209. doi:10.1002/aic.14974
41. Candelieri A, Perego R, Archetti F. Bayesian optimization of pump operations in water distribution systems. *J Glob Optim*. 2018;71(1):213-235. doi:10.1007/s10898-018-0641-2
42. Hutter F, Hoos HH, Leyton-Brown K. Sequential model-based optimization for general algorithm configuration. *International Conference on Learning and Intelligent Optimization*. Springer; 2011:507-523. doi:10.1007/978-3-642-25566-3_40
43. Hernández-Pérez H, Salazar-González J-J. A branch-and-cut algorithm for a traveling salesman problem with pickup and delivery. *Discret Appl Math*. 2004;145(1):126-139. doi:10.1016/j.dam.2003.09.013
44. DeMaio P and DesJardins Z. The bike sharing world map 2021 http://www.metrobikenet/.
45. Raviv T, Tzur M, Forma IA. Static repositioning in a bike-sharing system: models and solution approaches. *EURO J Transport Logistics*. 2013;2(3):187-229. doi:10.1007/s13676-012-0017-6
46. Raviv T, Kolka O. Optimal inventory management of a bike-sharing station. *IIE Trans*. 2013;45(10):1077-1093. doi:10.1080/0740817X.2013.770186
47. Chemla D, Meunier F, Calvo RW. Bike sharing systems: solving the static rebalancing problem. *Discret Optim*. 2013;10(2):120-146. doi:10.1016/j.disopt.2012.11.005
48. Dell'Amico M, Hadjicostantinou E, Iori M, Novellani S. The bike sharing rebalancing problem: mathematical formulations and benchmark instances. *Omega*. 2014;45:7-19. doi:10.1016/j.omega.2013.12.001
49. Erdoğan G, Battarra M, Calvo RW. An exact algorithm for the static rebalancing problem arising in bicycle sharing systems. *Eur J Oper Res*. 2015;245(3):667-679. doi:10.1016/j.ejor.2015.03.043
50. Schuijbroek J, Hampshire RC, Van Hoeve W-J. Inventory rebalancing and vehicle routing in bike sharing systems. *Eur J Oper Res*. 2017;257(3):992-1004. doi:10.1016/j.ejor.2016.08.029
51. Freund D, Norouzi-Fard A, Paul A, Wang C, Henderson SG, Shmoys DB. Data-driven rebalancing methods for bike-share systems. *Analytics for the Sharing Economy: Mathematics, Engineering and Business Perspectives*. Springer; 2020:255-278. doi:10.1007/978-3-030-35032-1_15
52. Chiariotti F, Pielli C, Zanella A, Zorzi M. A dynamic approach to rebalancing bike-sharing systems. *Sensors*. 2018;18(2):512. doi:10.3390/s18020512
53. Ghosh S, Varakantham P, Adulyasak Y, Jaillet P. Dynamic repositioning to reduce lost demand in bike sharing systems. *J Artif Intell Res*. 2017;58:387-430. doi:10.1613/jair.5308
54. Shui CS, Szeto WY. A review of bicycle-sharing service planning problems. *Transportation Research Part C: Emerging Technologies*. 2020;117:102648. doi:10.1016/j.trc.2020.102648
55. Helsinki Region Transport (HSL). City bike stations' Origin-Destination (OD) data. 2016. Accessed November 3, 2020. https://www.avoindata.fi/data/en_GB/dataset/helsingin-ja-espoon-kaupunkipyorilla-ajatut-matkat.
56. Kendall DG. Stochastic processes occurring in the theory of queues and their analysis by the method of the imbedded Markov chain. *Ann Math Stat*. 1953;24:338-354.
57. Kolmogoroff A. Über die analytischen methoden in der wahrscheinlichkeitsrechnung. *Math Ann*. 1931;104(1):415-458. doi:10.1007/BF01457949
58. Virtanen P, Gommers R, Oliphant TE, et al. Scipy 1.0: fundamental algorithms for scientific computing in Python. *Nat Methods*. 2020;17(3):261-272. doi:10.1038/s41592-019-0686-2
59. Forrester AIJ, Keane AJ, Bressloff NW. Design and analysis of "noisy" computer experiments. *AIAA J*. 2006;44(10):2331-2339. doi:10.2514/1.20068
60. Bouhlel MA, Hwang JT, Bartoli N, Lafage R, Morlier J, Martins JRRA. A Python surrogate modeling framework with derivatives. *Adv Eng Softw*. 2019;135:102662. doi:10.1016/j.advengsoft.2019.03.005
61. Jones DR. A taxonomy of global optimization methods based on response surfaces. *J Glob Optim*. 2001;21(4):345-383. doi:10.1023/A:1012771025575
62. Forrester AIJ, Sóbester A, Keane AJ. *Engineering design via surrogate modelling: a practical guide*. John Wiley & Sons; 2008. doi:10.1002/9780470770801
63. Helsinki Region Transport (HSL). Helsinki Region Transport's (HSL) city bicycle stations. 2016. Accessed November 3, 2020 https://www.avoindata.fi/data/en_GB/dataset/hsl-n-kaupunkipyoraasemat.
64. Kainu M. Helsingin kaupunkipyörät: Avoin data telinekohtaisista vapaiden pyörien määristä kausilta 2017, 2018, 2019, 2020 ja 2021. 2017. Accessed February 22, 2021 https://data.markuskainu.fi/opendata/kaupunkipyorat/.
65. Lee EH, Perrone V, Archambeau C, and Seeger M. Cost-aware Bayesian optimization. *ArXiv*. [Preprint] arXiv:200310870, 2020.
66. OpenStreetMap contributors. In: Anton AK, Edwin Z, Richard L, and Leyla Ö, eds. Planet dump. Retrieved from https://planet.osm.org. 2017. Accessed July 7, 2021. https://www.openstreetmap.org.

How to cite this article: Ikonen TJ, Heljanko K, Harjunkoski I. Surrogate-based optimization of a periodic rescheduling algorithm. *AIChE J*. 2022;68(6):e17656. doi:10.1002/aic.17656