Article

A General Cooperative Optimization Approach for Distributing Service Points in Mobility Applications

Thomas Jatschka [1,*], Günther R. Raidl [1] and Tobias Rodemann [2]

[1] Institute of Logic and Computation, TU Wien, 1040 Vienna, Austria; raidl@ac.tuwien.ac.at
[2] Honda Research Institute Europe, 63073 Offenbach am Main, Germany; tobias.rodemann@honda-ri.de
* Correspondence: tjatschk@ac.tuwien.ac.at

Abstract: This article presents a cooperative optimization approach (COA) for distributing service points for mobility applications, which generalizes and refines a previously proposed method. COA is an iterative framework for optimizing service point locations, combining an optimization component with user interaction on a large scale and a machine learning component that learns user needs and provides the objective function for the optimization. The previously proposed COA was designed for mobility applications in which single service points are sufficient for satisfying individual user demand. This framework is generalized here for applications in which the satisfaction of demand relies on the existence of two or more suitably located service stations, such as in the case of bike/car sharing systems. A new matrix factorization model is used as surrogate objective function for the optimization, allowing us to learn and exploit similar preferences among users w.r.t. service point locations. Based on this surrogate objective function, a mixed integer linear program is solved to generate an optimized solution to the problem w.r.t. the currently known user information. User interaction, refinement of the matrix factorization, and optimization are iterated. An experimental evaluation analyzes the performance of COA with special consideration of the number of user interactions required to find near optimal solutions. The algorithm is tested on artificial instances, as well as instances derived from real-world taxi data from Manhattan. Results show that the approach can effectively solve instances with hundreds of potential service point locations and thousands of users, while keeping the user interactions reasonably low. A bound on the number of user interactions required to obtain full knowledge of user preferences is derived, and results show that with 50% of performed user interactions the solutions generated by COA feature optimality gaps of only 1.45% on average.

Keywords: heuristic optimization; location planning; cooperative optimization; preference learning

check for updates

Citation: Jatschka, T.; Raidl, G.R.; Rodemann, T. A General Cooperative Optimization Approach for Distributing Service Points in Mobility Applications. *Algorithms* **2021**, *14*, 232. https://doi.org/10.3390/a14080232

Academic Editor: Frank Werner

Received: 28 June 2021
Accepted: 5 August 2021
Published: 6 August 2021

1. Introduction

A fundamental ingredient for optimizing the locations of service points in mobility applications, such as charging stations for electric vehicles or pickup and drop-off stations for car/bike sharing systems, is the distribution of existing customer demand to be potentially fulfilled in the considered geographical area. While there exists a vast amount of literature regarding setting up service points for mobility applications, such as vehicle sharing systems [1–4] or charging stations for electric vehicles [5–8], estimations of the existing demand distribution are usually obtained upfront by performing customer surveys, considering demographic data, information on the street network and public transport, and not that seldom including human intuition and political motives. However, such estimations are frequently imprecise and a system built on such assumptions might not perform as effectively as it was originally hoped for. For example, in Reference [9], GPS-based travel survey data of fossil fueled cars is used for setting up charging stations for electric vehicles. However, as pointed out by Pagany et al. [10], it cannot be assumed that the driving behavior of customers remains unchanged when switching from fossil fueled cars to electric vehicles. Furthermore, Pagany et al. [10] present a survey of 119 publications for locating charging stations for electric vehicles in which they also discuss further problems with the above mentioned demand estimation methods.

A more frequent usage of a service system by a customer will in general depend not only on the construction of a single service point on a particular location but more globally on non-trivial relationships of the customer's necessities and preferences in conjunction with larger parts of the whole service system. For example, in the case of bike/car sharing systems, a well placed rental station close to the origin of a trip might be worthless if there does not also exist a suitable location near the destination for returning the vehicle. Furthermore, some customers might use multiple modes of transport for a single trip [11]. Consequently, some more distant service station for returning the vehicle might be acceptable if this place is well connected by public transport used for an additional last leg [12]. Thus, there also might be alternatives for fulfilling demand that cannot all be exactly specified by potential users. The example with an additional leg by public transport also illustrates that geographical closeness is not always the deciding factor.

To address these issues, Jatschka et al. [13] proposed the concept of a *Cooperative Optimization Algorithm* (COA) which, instead of estimating customer demand upfront, directly incorporates potential users in the location optimization process by iteratively confronting them with location scenarios and asking for evaluation. Based on the user feedback, a machine learning model is trained which is then used as surrogate objective function to evaluate solutions in an optimization component. User interaction, a corresponding refinement of the surrogate objective function, and the optimization are iterated. Expected benefits of such a cooperative approach are a faster and easier data acquisition [14], the direct integration of users into the whole location planning process, a possibly stronger emotional link of the users to the product, and ultimately better understood [15] and more accepted optimization results [14]. Potential customers further know local conditions and their particular properties, including also special aspects that are not all easily foreseen in a classical demand acquisition approach. To the best of our knowledge, such an approach has not yet been considered in the area of locating service points for mobility applications.

As the initial proof-of-concept approach in Reference [13] did not scale well to larger application scenarios, we described a more advanced cooperative optimization approach relying on matrix factorization in a preliminary conference paper [16]. The current article deepens and extends this work, particularly in the following aspects.

We now apply a more advanced matrix factorization originally introduced in Reference [17], which allows us to exploit that only a small fraction of locations is actually relevant for each user and that user data is not missing at random.

Moreover, the COA in Reference [16] considered only mobility applications in which single service points are sufficient for satisfying a particular demand of a user, such as the placement of charging stations for electric vehicles. The main contribution of this paper is to extend the solution approach towards applications in which the satisfaction of demand typically relies on the existence of suitable pairs or, more generally, tuples of service stations, such as in the case of bike or car sharing systems where a vehicle is first picked up at a rental station near the origin and returned at a station within easy reach of the destination.

This extended framework is tested on different artificial benchmark scenarios using an idealized simulation of user interactions. The benchmark scenarios were hereby created in a controlled way so that optimal reference solutions could be obtained for comparison. One group of benchmark scenarios was derived from real world taxi trip data from Manhattan. To characterize the amount of user interaction COA requires, an upper bound on the number of necessary non-redundant user interactions for obtaining full knowledge of user preferences is derived as a baseline. Results on our instances show that solutions generated by COA feature optimality gaps of only 1.45%, on average, with 50% of performed user interactions.

The next section discusses related work. In Section 3, we introduce our formal problem setting, the *Generalized Service Point Distribution Problem (GSPDP)* , which reflects the essence of various location problems for mobility services. Afterwards, in Section 4, we detail the cooperative optimization framework for solving the GSPDP. Section 5 describes how benchmark scenarios for testing were generated, and, in Section 6, we present and discuss experimental results. Section 7 concludes this work with an outlook on promising future work.

2. Related Work

The considered problem in general falls into the broad category of facility location problems, i.e., optimization problems where facilities should be set up on a subset of given potential locations as economically as possible in order to fulfill a certain level of customer demand. Similar to a p-median or maximal covering location model, as in Reference [18], we limit the facilities to be opened. However, instead of imposing a direct limit on the number of facilities, in our formulation, a maximum budget for setting up facilities is specified, and each facility is associated with setup costs. In general, our problem can be classified as an uncapacitated fixed charge Facility Location Problem (FLP), as in Reference [19], however, without explicitly given demands (see below). Moreover, our problem also has similarities to stochastic or robust FLPs [20] in which problem inputs, such as the demand, may be uncertain. For such problems, uncertain inputs are usually modeled via random variables [21] or scenario-based approaches [22]. For a survey on FLPs, see Reference [23].

In our problem formulation, we have mobility applications in mind, such as the distribution of charging stations for electric vehicles or setting up rental stations for car/bike sharing. While there already exists a vast amount of literature for setting up such systems, to the best of our knowledge, all existing work essentially assumes customer demand to be estimated upfront. For example, Refs. [8,24] use parking information to identify promising locations for electric vehicle charging stations. Ref. [25] locates charging stations for an on-demand bus system using taxi probe data of Tokyo. Moreover, census data are commonly used for estimating demand for car sharing systems [26], bike sharing systems [27], or for setting up electric vehicle charging stations [28]. Data mining and other related techniques are also often employed to detect traffic patterns for bike sharing systems [29,30], as well as car sharing systems [31].

Ciari et al. [32] recognize the difficulties in making demand predictions for new transport options based on estimated data and, therefore, propose to use an activity-based microsimulation technique for the modeling of car sharing demand. The simulation is completed with help of the travel demand simulator MATSim [33].

There also exist some works that take user preferences into account, e.g., in Reference [34], a car sharing system is designed based on different assumptions on the behavior of users. The authors come to the conclusion that providing real-time information to customers can greatly improve the service level of a system if users are willing to visit a different station if their preferred one does not have a vehicle available.

In our approach, we substantially deviate from this traditional way of acquiring existing demand upfront and instead resort to an interactive approach. Potential future customers are directly incorporated in the optimization process as an integral part by iteratively providing feedback on meaningfully constructed location scenarios. In this way, we learn user demands on-the-fly and may avoid errors due to unreliable a priori estimations. For a survey on interactive optimization algorithms, in general, see Reference [14]. The performance of interactive algorithms is strongly influenced by the quality of the feedback given by the interactors. Too many interactions with a user will eventually result in user exhaustion [35], negatively influencing the reliability of the obtained feedback. Additionally, interacting with users can be quite time consuming, even when dealing with a single user. Hence, in order to keep the interactions with users low, one can resort to surrogate-based optimization approaches [36,37]. Surrogate models are typically machine learning models serving as proxy of functions that are difficult to evaluate [38]. In Reference [16], as well as in this contribution, we make use of a matrix factorization [39]-based surrogate model. Matrix factorization is a collaborative filtering technique which is frequently used in recommender systems; see, e.g., Reference [40].

As already pointed out, the basic concept of COA was already presented in Reference [13], where we made use of an adaptive surrogate model [41]. The underlying structure of this surrogate model is formed by a large set of individual smaller models, i.e., one machine learning model for each combination of user and potential service point location, which are trained with the feedback of the respective users. While initially being instantiated by simple linear models, these machine learning models are step-wise upgraded as needed to more complex regressors during the course of the algorithm in order to cope with possibly encountered higher complexity. Specifically, in Reference [13], each linear model can be upgraded to a neural network. The number of neurons in the hidden layer is increased whenever the training error exceeds a certain threshold. Overfitting is effectively avoided by not making the individual models unnecessarily large. A Variable Neighborhood Search (VNS) [42] was used as optimization core to generate new solutions w.r.t. the current surrogate function. In Reference [43], the performance of the VNS optimization core was compared to an optimization core using a population-based iterated greedy approach [44]. Unfortunately, this first realization of COA exhibits severe limitations in the scalability to larger numbers of potential service point locations and/or users, particularly as all users are considered independently of each other.

The current work builds upon the observation that, in a larger user base, there are typically users sharing the same or similar needs or preferences. Identifying these shared demands and exploiting them to improve scalability, as well as to reduce the required feedback per user, is a main goal here. While the basic principles of COA remain the same, major changes are performed in the way the approach interacts with users and how the feedback of users is processed, as well as how new candidate solutions are generated. Moreover, the adaptive surrogate model from Reference [13] realized by the large set of underlying simpler machine learning models is replaced by a single matrix factorization-based model that is able to exploit said similarities between users. Besides our aforementioned preliminary conference paper already sketching the application of a matrix factorization within COA [16], to the best of our knowledge, there exists no further work on interactive optimization approaches for location planning in mobility applications.

3. The Generalized Service Point Distribution Problem

The *Generalized Service Point Distribution Problem* (GSPDP) considered here is an extension of the *Service Point Distribution Problem* (SPDP) introduced in Reference [16]. Given are a set of locations $V = \{1, \ldots, n\}$, at which service points may be set up, and a set of potential users $U = \{1, \ldots, m\}$. The fixed costs for establishing a service point at location $v \in V$ are $z_v^{\text{fix}} \geq 0$, and this service point's maintenance over a defined time period is supposed to induce variable costs $z_v^{\text{var}} \geq 0$. The total setup costs of all stations must not exceed a maximum budget $B > 0$. Furthermore, it is assumed that opened service stations are able to satisfy an arbitrary amount of customer demand. For each unit of satisfied customer demand, a prize $q > 0$ is earned.

A solution to the GSPDP is a subset $X \subseteq V$ of all locations where service points are to be set up. A solution X is feasible if its total fixed costs do not exceed the maximum budget B, i.e.,

$$z^{\text{fix}}(X) = \sum_{v \in X} z_v^{\text{fix}} \leq B. \qquad (1)$$

Given the set of users U, we assume that each user $u \in U$ has a certain set of *use cases* C_u, such as going to work, to a recreational facility, or shopping. Each use case $c \in C_u$ is associated with a demand $D_{u,c} > 0$ expressing how often the use case is expected to be frequented by user u within some defined time period, such as a week or a month. For each unit of satisfied customer demand, a prize $q > 0$ is earned. The demand of each use case can possibly be satisfied by different service points or subsets of service points to different degrees, depending on the concrete application and the customer's preferences. Note that use cases are here just labels and are not directly associated with specific geographic locations. This separation is intentionally done in order to keep flexibility: some use cases, such as shopping or the visit of a fitness center, may possibly be realized at different places, and, as already mentioned, occasionally, a service station farther away from a specific target location may also be convenient if some other mode of transportation is used as additional leg.

Depending on the actual application and characteristics of a use case, demand may be fulfilled by a single service station, e.g., when charging batteries of an electric vehicle, or a suitable combination of multiple service stations may be needed, such as when renting a vehicle at one place and returning it somewhere else. While Reference [16] just considered the first case, we pursue here the general case of possibly requiring multiple service points to fulfill demand for a single use case.

To model this aspect formally, we associate each use case c of a user u with a set of *Service Point Requirements* (SPR) $R_{u,c}$. Similar to use cases, these SPRs are not directly associated with geographic locations but are an abstract entity, such as "place within easy reach of home to rent a vehicle" or "place close to a supermarket to return a vehicle", with which a user can express the dependency on multiple service points to fulfill the needs of one use case. Thus, the demand of such a use case can only be satisfied if a service point exists at a suitable location for each of the use case's SPRs. Note that multiple use cases of a user may also share the same SPR(s). For example, a use case referring to a trip from home to work and one from home to a supermarket may share the SPRs "place within easy reach of home to rent a vehicle". The set of all different SPRs over all use cases of a user u is denoted by $R_u = \bigcup_{c \in C_u} R_{u,c}$. Moreover, let $R = \bigcup_{u \in U} R_u$ be the set of all SPRs over all users. Note that, in this notation, different users never share the same SPR labels, although labels may refer to similar SPRs.

For indicating how suitable a location is w.r.t. to an SPR, we define values $w_{r,v} \in [0,1]$, indicating the suitability of a service point at location $v \in V$ to satisfy the needs of user $u \in U$ concerning SPR $r \in R_{u,c}$ in the use case $c \in C_u$. A value of $w_{r,v} = 1$ represents perfect suitability, while a value of zero means that location v is unsuitable; values in between indicate partial suitability.

With these suitability values in mind, the objective of the GSPDP is to maximize

$$f(X) = q \cdot \sum_{u \in U} \sum_{c \in C_u} D_{u,c} \cdot \min_{r \in R_{u,c}} \left(\max_{v \in X} w_{r,v} \right) - \sum_{v \in X} z_v^{\text{var}}. \tag{2}$$

In the first term of this objective function, the obtained prize for the expected total satisfied demand is determined by considering for each user u, each use case c, and each SPR r a most suitable location $v \in V$ at which a service point is to be opened ($v \in X$). Over all SPRs of a use case, the minimum of the obtained suitability values is taken so that the full demand is only fulfilled when, for each SPR, an ideally suited service station is planned, and no demand is fulfilled as soon as one of the SPRs does not have an appropriate service point. The second term of the objective function represents the total maintenance costs for the service stations.

By linearizing the above objective function, the GSPDP can be formulated as a mixed integer linear program (MILP) with the following variables. Binary variables x_v indicate whether or not a service point is deployed at location $v \in V$, i.e., the binary vector $x = (x_v)_{v \in V}$ is the incidence vector of a corresponding solution $X \subset V$. Additional variables $h_{r,v}$ are used to indicate the actually used location $v \in V$ for each SPR $r \in R$. The degree to which a use case $c \in C_u$ of a user $u \in U$ can be satisfied is expressed by continuous variables $y_{u,c} \in [0,1]$. The GSPDP is then stated as follows.

$$\max \quad q \cdot \sum_{u \in U} \sum_{c \in C_u} D_{u,c}\, y_{u,c} - \sum_{v \in V} z_v^{\text{var}}\, x_v, \tag{3}$$

$$\sum_{v \in V} z_v^{\text{fix}}\, x_v \leq B, \tag{4}$$

$$\sum_{v \in V} h_{r,v} \leq 1 \qquad \forall r \in R, \tag{5}$$

$$\sum_{v \in V} w_{r,v} \cdot h_{r,v} \geq y_{u,c} \qquad \forall u \in U,\ c \in C_u,\ r \in R_{u,c}, \tag{6}$$

$$h_{r,v} \leq x_v \qquad \forall v \in V,\ r \in R, \tag{7}$$

$$x_v \in \{0,1\} \qquad \forall v \in V, \tag{8}$$

$$0 \leq y_{u,c} \leq 1 \qquad \forall u \in U,\ c \in C_u, \tag{9}$$

$$0 \leq h_{r,v} \leq 1 \qquad \forall r \in R,\ v \in V. \tag{10}$$

In correspondence to the definition of f, the objective value is calculated in (3) as the sum of the prizes earned for fulfilled demand minus the costs for opening service stations. Inequality (4) ensures that the budget is not exceeded. Inequalities (5) ensure that, at most, one location is selected for each SPR. As our objective is to maximize the revenue, it is ensured that always a suitable service point location with the highest suitability value for each SPR is chosen. Inequalities (6) determine the degrees to which the use cases are satisfied, considering that the actually fulfilled demand of a use case is assumed to be proportional to the minimum suitability value of the locations selected for the SPRs of the use case. Last but not least, Inequalities (7) ensure that only locations at which service points are to be opened can be used for SPRs and, thus, to satisfy demand of use cases. The size of this model in terms of the number of variables, as well as the number of constraints, is in $\mathcal{O}(m\, n\, |R|)$.

Theorem 1. *The GSPDP is NP-hard.*

Proof. NP-hardness of the GSPDP is proven by providing a reduction from the well known NP-hard Maximal Covering Location Problem (MCLP) [45] in the variant stated by Farahani et al. [46]. Given are a set of possible facility locations $\mathcal{J}$, a maximum number p of facilities to be opened, and a set of demand nodes $\mathcal{D}$. Moreover, each demand node $i \in \mathcal{D}$ is associated with a demand $a_i \geq 0$ and a subset of facilities $\mathcal{F}_i \subseteq J$ of which each is able to cover the node's full demand. The goal of the MCLP is to select up to p locations for opening facilities in order to maximize the total demand covered.

Given an instance to the MCLP, we construct a corresponding GSPDP instance, in which the set of locations V corresponds to the set of facilities $\mathcal{J}$, and the set of users U corresponds to the set of demand nodes $\mathcal{D}$. Moreover, each user $u \in U$ only has a single use case with a single SPR and demand a_i with $u = i$. Building costs z_v^{fix} for a location $v \in V$ are set to one, while the maintenance costs z_v^{var} are zero. The budget of the GSPDP instance is set to p, and the prize for a unit of covered demand q is set to one. The suitability value $w_{r,v}$ is set to one for $v \in V$ and $r \in R$ if facility i can satisfy the demand of demand node j, i.e., $j \in \mathcal{F}_i$, and zero otherwise.

Let (x, y, h) be a feasible solution to this derived GSPDP instance. A corresponding feasible solution to the MCLP is obtained by opening facilities at all locations $j \in \mathcal{J}$ for which $x_v = 1$. Due to the budget constraint (4), at most, p facilities are opened in the MCLP instance; thus, there is a bijective mapping of feasible GSPDP solutions to feasible MCLP solutions.

Since each user in the GSPDP instance only has one use case, and each use case only consists of one SPR, the sets U, C, and R all contain the same elements. By our definitions, variables $y_{u,c}$ indicating the covered SPRs, therefore, also indicate the covered demand nodes of the MCLP instance. More generally, we also have a bijective mapping of covered SPRs in the GSPDP instance to covered demand nodes in the MCLP instance. Last but not least, due to our definitions of the suitability values $w_{r,v}$, the fixed and variable costs for opened stations, and the prize per unit of fulfilled demand, the objective values of corresponding GSPDP and MCLP solutions also correspond. Since all applied transformations require polynomial time, it follows that the GSPDP is NP-hard. □

In the next section, we present a cooperative algorithm for solving the GSPDP when no a priori information about the suitability values $w_{r,v}$ of potential locations is known but can only be obtained by querying the users in a limited fashion.

4. The Cooperative Optimization Algorithm

A crucial aspect for solving the GSPDP is that determining the suitability values $w_{r,v}$ is no trivial task. While a user may be able to list a small number of best suited service station locations for each of his/her SPRs, we have to consider it practically infeasible to obtain reasonably precise suitability values for all potential service station locations V of each SPR by directly asking the users. A complete direct questioning would not only be extremely time consuming, but users would easily be overwhelmed by the large number of possibilities, resulting in incorrect information. For example, users easily tend to only rate their preferred options as suitable and might not consider certain alternatives as also feasible, although they actually might be, on second thought, when no other options are available. The problem of user fatigue substantially impacting the quality of obtained feedback when too much information is asked from a user is, for example, discussed in Reference [35].

Hence, interaction with users needs to be kept to a minimum and should be done wisely to extract as much meaningful information as possible. Moreover, users must be confronted with easy questions whose answers at the same time provide strong guidance for the target system. Based on this philosophy, we present a *Cooperative Optimization Algorithm (COA)* for solving the GSPDP if no a priori information about the use cases of the users, their respective demands, or the suitability values is known. The general concept of COA was originally introduced in Reference [13]. In this paper, this framework is substantially adapted and extended towards better scalability and for solving instances of the more general GSPDP, as specified in the previous section.

In Section 4.3, we formally define how users can interact with COA. Note, however, that COA does not put a strict limit on the number of allowed interactions with each user. Instead, we will measure the effectivity of our COA framework by the number of user interactions the framework requires for generating a (close to) optimal solution.

4.1. Methodology

In this subsection, a high level overview of COA is given before detailing the individual components in successive subsections. Figure 1 shows the basic methodology: In each iteration, the algorithm first generates location scenarios for a subset of users to evaluate. Based on the users' ratings of the scenarios, a surrogate objective function is continuously updated over the iterations. The GSPDP instance with the current surrogate objective function is then solved, yielding a solution. In the next iteration, this solution is a basis for deriving further meaningful location scenarios to be presented to users again. The surrogate objective function, thus, learns to represent the users' needs, more specifically the suitability values $w_{r,v}$, better and better, and the solutions obtained from the optimization will become more precise over time.

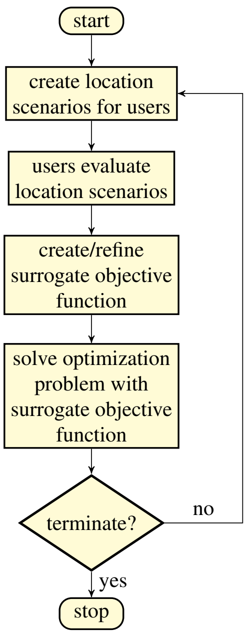
start
create location scenarios for users
users evaluate location scenarios
create/refine surrogate objective function
solve optimization problem with surrogate objective function
terminate?
no
yes
stop

Figure 1. Basic methodology of the COA framework.

Figure 2 exemplifies how the evaluation of a location scenario may look from the perspective of a user.

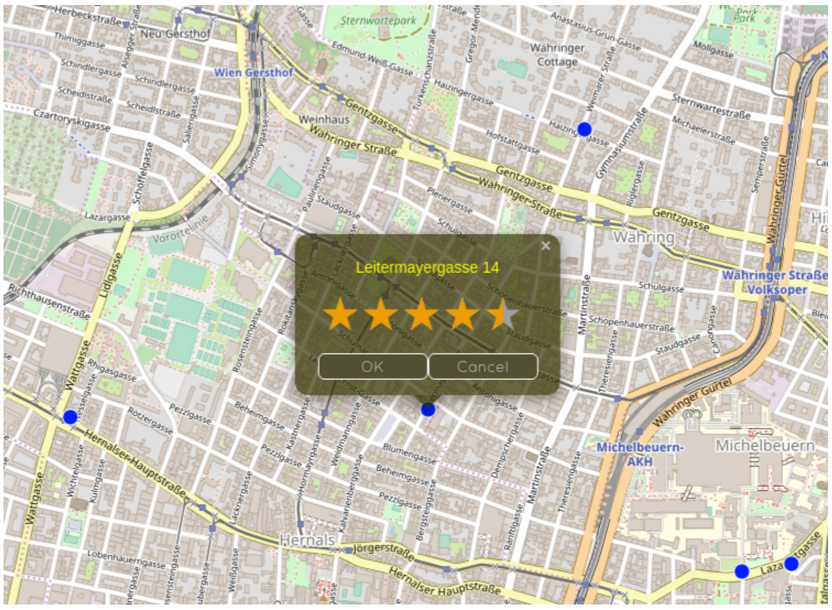

Figure 2. Exemplary evaluation of a location scenario by a user in the COA framework: A potential solution in form of a map with highlighted locations is presented, and the user can than provide suitability values for the highlighted locations in the form of ratings.

From a technical point-of-view, the COA framework consists of a *Feedback Component (FC)* , an *Evaluation Component (EC)*, an *Optimization Component (OC)*, and a *Solution Management Component (SMC)*.

First, the FC is called, starting an initialization phase by asking each user $u \in U$ to specify the user's use cases C_u and associated SPRs $R_{u,c}$, as well as corresponding demands $D_{u,c}$, $c \in C_u$. Afterwards, the FC is responsible for collecting information from the user, i.e., users can interact with the framework at this stage of the algorithm. User information is collected by generating individual location scenarios for each user which are presented to the user in order to obtain his/her feedback. A user $u \in U$ then has to provide suitability values $w_{r,v}$ for locations $v \in S$ of solution scenarios S presented to him with respect to a use case requirement $r \in R_u$.

The feedback obtained from the users is processed in the EC. The EC maintains and continuously updates a *surrogate suitability function* $\tilde{w}_\Theta(r,v)$ approximating the suitability values $w_{r,v}$ of service point locations $v \in V$ w.r.t. SPR $r \in R$ without interacting with the respective user. This function is realized by a machine learning model with parameters (weights) Θ. Based on this surrogate function, the EC also provides the surrogate objective function

$$\tilde{f}_\Theta(X) = q \cdot \sum_{u \in U} \sum_{c \in C_u} D_{u,c} \cdot \min_{r \in R_{u,c}} \left(\max_{v \in X} \tilde{w}_\Theta(r,v) \right) - \sum_{v \in X} z_v^{\text{var}}, \tag{11}$$

with which a candidate solution X can be approximately evaluated. This fast approximate evaluation is particularly important for intermediate candidate solutions generated during the optimization process which are not evaluated by the users. The respective learning mechanism of the surrogate objective function is also part of the EC.

A call of the OC is supposed to determine an optimal or close-to-optimal solution to the problem with respect to the EC's current surrogate objective function $\tilde{f}_\Theta$. Note that the surrogate objective function never changes during a call of the OC, only in each major iteration of the framework after having obtained new user feedback. With the exception of the first call, the optimization procedure of the OC is warm-started with the current best solution $\tilde{X}^*$ as initial solution to possibly speed up the optimization process.

Finally, the SMC efficiently stores and manages information on all candidate solutions that are relevant for more than one of the above components and, particularly, also the location scenario evaluations provided by the users so far, as well as the solutions X_{OC} returned by the OC.

The whole process is repeated until some termination criterion is reached, e.g., the discrepancy of user feedback and the results of the EC is small enough or a maximum number of user interactions has been reached. In the end, COA returns a solution with the highest surrogate objective value of all of the so far generated solutions.

Figure 3 and Algorithm 1 give a summary of the whole COA procedure and of the main tasks of each of the components of COA.

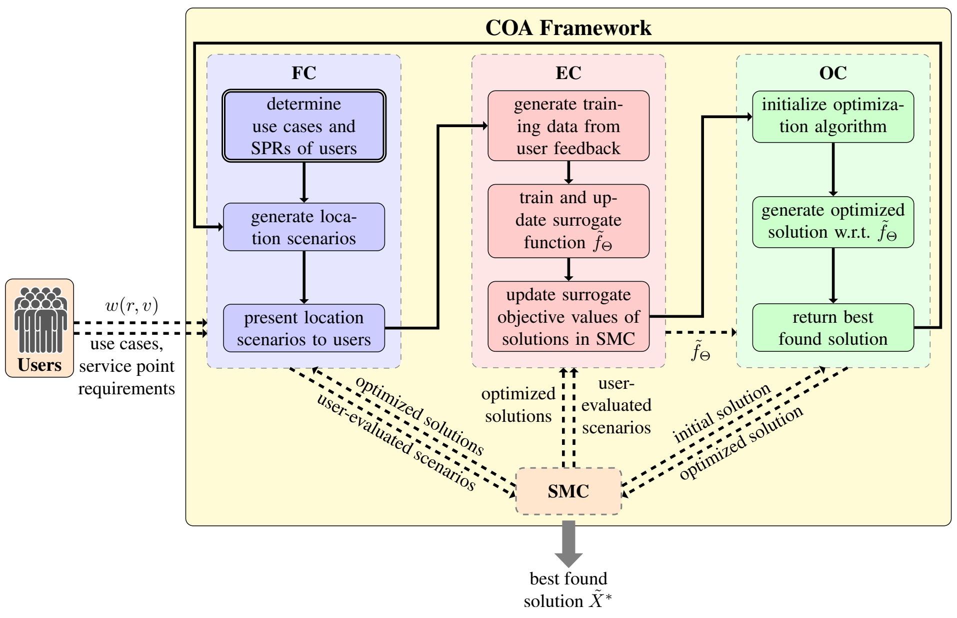
COA Framework
FC
determine use cases and SPRs of users
generate location scenarios
present location scenarios to users
EC
generate training data from user feedback
train and update surrogate function $\tilde{f}_\Theta$
update surrogate objective values of solutions in SMC
OC
initialize optimization algorithm
generate optimized solution w.r.t. $\tilde{f}_\Theta$
return best found solution
Users
$w(r,v)$
use cases, service point requirements
$\tilde{f}_\Theta$
optimized solutions
user-evaluated scenarios
optimized solutions
user-evaluated scenarios
initial solution
optimized solution
SMC
best found solution $\tilde{X}^*$

Figure 3. Components of COA and their interaction. The framework consists of the feedback component (FC), the evaluation component (EC), the optimization component (OC), and the solution management component (SMC). Users interact with the framework via the FC.

In the next subsections, we describe each component's functionality in more detail.

Algorithm 1: Basic Framework

```
    Input: an instance of the GSPDP
    Output: best solution $\tilde{X}^* \subseteq V$ found
1:  Feedback Component:
2:      for $u \in U$ do
3:          obtain use cases $C_u$ with associated demands $D_{u,c}$ and service point
              requirements $R_{u,c}$, $\forall c \in C_u$, from user $u$;
4:      end for
5:  while no termination criterion satisfied do
6:      Feedback Component:
7:          for each scenario generation strategy $\mathcal{S}_r$ do
8:              $R' \leftarrow$ random sample of $R$;
9:              for $r \in R'$ do
10:                 $S_r \leftarrow$ generate set of location scenarios according to strategy $\mathcal{S}_r$;
11:                 present scenario to the corresponding user;
12:                 update SMC with ratings obtained from $\mathcal{S}_r$;
13:             end for
14:         end for
15:     Evaluation Component:
16:         train surrogate model $\tilde{w}_\Theta$, yielding updated surrogate obj. function $\tilde{f}_\Theta$;
17:         re-evaluate all solutions stored in the SMC with new $\tilde{f}_\Theta$;
18:     Optimization Component:
19:         $X_{\mathrm{OC}} \leftarrow$ generate optimal solution w.r.t. the EC's $\tilde{f}_\Theta$;
20:         update SMC with $X_{\mathrm{OC}}$;
21: end while
22: return overall best found solution $\tilde{X}^*$;
```

4.2. Solution Management Component

The SMC stores and manages so far considered solutions and evaluations by the users. The set of solutions obtained from the OC in all major iterations is stored in a set we denote by $\mathcal{X}$. For each solution $X \in \mathcal{X}$, the SMC keeps track of its current surrogate objective value $\tilde{f}_\Theta(X)$. Hence, the surrogate objective values of the solutions in $\mathcal{X}$ are updated in each major COA iteration whenever the EC updates the surrogate suitability function. The current best solution in $\mathcal{X}$, i.e., the solution with the highest surrogate objective value, is denoted by $\tilde{X}^*$.

All feedback obtained from the users via the presented location scenarios is collected and stored in the SMC in a hash map: Its set of keys, which we denote by K, is the set of pairs (r, v) with $r \in R$, $v \in V$ for which suitability values $w_{r,v}$ have been obtained from the users, and the respective values are the $w_{r,v}$.

Last but not least, through the FC, we are also able to obtain upper bounds on suitability values $w_{r,v}$, with $v \in V$, $r \in R$, as will be explained in Section 4.3. These upper bounds are stored in the SMC as $w^{\mathrm{UB}}_{r,v} \in [0, 1]$.

4.3. Feedback Component

The FC is responsible for extracting as much information as possible from the users with as few interactions as necessary in order to not to fatigue them.

In the context of the COA framework, user interactions are understood as letting a user evaluate (a small number of) *location scenarios*. Similar to a solution, a location scenario is a subset of locations in V at which service stations are assumed to be opened. In contrast to a solution, the budget constraint (1) does not need to be respected, and a location scenario may, therefore, include an arbitrary number of service points. Recall Figure 2 for an example of how an interaction with the COA framework may look from the perspective of a user.

Let $S \subseteq V$ be such a location scenario provided to a user u with respect to one of the user's SPRs $r \in R_u$. It is then assumed that the user returns as evaluation of the scenario S either a best suited location $v_{r,S} \in S$ and the corresponding suitability value $w(r, v_{r,S}) > 0$ or the information that none of the locations of the scenario S is suitable. The latter case implies that $w_{r,v} = 0$ for all $v \in S$. In case multiple locations are equally well suited, we assume that the user selects one of them at random. It is assumed here that the suitability of a location w.r.t. an SPR can be specified by the user on a five-valued scale from zero, i.e., completely unsuitable, to one, i.e., perfectly suitable; a more fine grained evaluation would not make much practical sense.

Clearly, this definition of user interaction is simplified and idealized, particularly as we assume here that all users always give precise answers. In a real application, the uncertainty of user feedback and the possibility of misbehaving users who intentionally give misleading answers also need to be considered among other aspects. Moreover, it would be meaningful to extend the possibilities of user feedback. For example, users could be allowed to optionally rate more than one suitable locations for an SPR in one scenario or to make suggestions which locations to additionally include in a scenario as we considered it in Reference [13].

In each iteration of COA, users are presented with individual sets of location scenarios $\mathcal{S}_r$ for their service point requirements $r \in R$. These scenarios are compiled according to the following strategies.

Remember that location suitability values obtained from the users are later used in the EC for training the surrogate function $\tilde{w}_\Theta$. Moreover, by enforcing that each user is required to select a best suited service point location in a presented location scenario for an SPR r, a suitability value indicated by the user for some location $v_{r,S}$ also serves as upper bound on the suitability values of all other locations in the location scenario S; thus, $w_{r,v_{r,S}} \geq w_{r,v}$, $\forall v \in S$. By $w^{\mathrm{UB}}_{r,v}$, the SMC maintains for each SPR $r \in R$, and each location $v \in V$ the so far best obtained upper bound on each $w_{r,v}$; initially, $w^{\mathrm{UB}}_{r,v} = 1$.

Let $V_w(r) = \{v \mid w(r,v) > 0\}$ be the initially unknown set of locations that are actually relevant to a user $u \in U$ w.r.t. an SPR $r \in R_u$. A straightforward strategy to identify this set is to iteratively present the user scenarios $S^{\mathrm{V}}_r = \{v \in V \mid (r,v) \notin K\}$, containing all locations $v \in V$ for which no entry $(v,r) \in K$ exists yet, i.e., locations for which no suitability values are known yet w.r.t. r. Following this strategy, it can be ensured to identify a new location of $V_w(r)$ in every iteration of COA. Note, however, that it can only be guaranteed that $V_w(r)$ is completely known once the user returns that none of the locations in the last scenario S^{V}_r are suitable for r. Consequently, $V_w(r)$ will be completely known after $|V_w(r)| + 1$ user interactions.

Hence, an upper bound I^{UB}_u on the total number of required interactions with user u for completely identifying all relevant locations for all of his/her use cases is

$$I^{\mathrm{UB}}_u = \sum_{r \in R_u} (|V_w(r)| + 1). \qquad (12)$$

While this value is unknown in a real-world scenario, it allows us to establish a measure of quality on how well our strategy for presenting scenarios to users performs within our testing environments.

The same combination of two strategies for generating scenarios w.r.t. an SPR $r \in R$ as in Reference [16] is used. The first strategy generates scenarios according to the approach described above, i.e, $S^{\mathrm{V}}_r = \{v \in V \mid (r,v) \notin K\}$. The second strategy generates scenarios $S^*_r = \{v \in \tilde{X}^* \mid (r,v) \notin K\}$ containing all locations from the current best solution that have not been rated yet w.r.t. r.

Note that, for users, generally only a fraction of the service point locations in V is actually relevant to one of their SPRs. Hence, when presenting a user $u \in U$ two scenarios for each of the user's SPRs, every iteration, the number of user interactions would quickly exceed I_u^{UB}. Therefore, in the first COA iteration, a scenario S_r^{V} is generated for each $r \in R$, but, in successive iterations, scenarios are only generated for subsets of R. More specifically, from the second iteration onward, ς^{V} and ς^* percent of the SPRs $R_K = \{r \in R \mid \exists v \in V : (r, v) \notin K\}$ are randomly selected for generating scenarios according to S_r^{V} and S_r^*, respectively, with ς^{V} and ς^* being strategy parameters.

4.4. Evaluation Component

The EC processes the user feedback obtained from the FC and provides the means for evaluating candidate solutions without relying on users in particular within the OC. The exact objective function f from (2), which is based on the mostly unknown suitability values $w_{r,v}$ with $r \in R$, $v \in V$, is approximated by the surrogate objective function $\tilde{f}_\Theta$, cf. (11), making use of the following *surrogate suitability function*

$$\tilde{w}_\Theta(r, v) = \begin{cases} w_{r,v} & \text{if } (r, v) \in K \\ \max(0, \min(w_{r,v}^{\mathrm{UB}}, g_\Theta(r, v))) & \text{else.} \end{cases} \tag{13}$$

Generally speaking, g_Θ is here a learnable function with weight parameters Θ approximating $w_{r,v}$ for all unknown pairs $(r, v) \notin K$. The above definition, thus, ensures that $\tilde{w}_\Theta$ always returns known values $w_{r,v}$ and otherwise respects lower bounds zero and upper bounds $w_{r,v}^{\mathrm{UB}}$, giving function g more freedom. Upper bounds $w_{r,v}^{\mathrm{UB}}$ are initially set to one. In Section 4.3, discussion is given on how tighter upper bounds are derived.

Suitability values are approximated by exploiting similarities of SPRs among users. In general, we cannot expect that there exist users having the same needs in all respects, i.e., the users have the very same use cases with the same demands. However, given a sufficiently large user base, it is realistic that there are users having similar SPRs and associated preferences concerning suitable locations.

A popular collaborative filtering technique for exploiting similarities among user preferences is *matrix factorization* [39], which we also apply here. Given an incomplete matrix containing ratings $\mathcal{R} = (w_{i,j})_{i \in U, j \in P}$ for a set of users U over a set of products P, the idea behind matrix factorization is to decompose this matrix into two smaller matrices, a user/feature matrix ξ and a product/feature matrix ν, such that the product of these two matrices approximates the original matrix. An unknown rating, i.e., a rating not contained in the original matrix $\mathcal{R}$, can then be estimated as the dot product of the corresponding feature vectors in matrix ξ and matrix ν, respectively.

Moreover, we also want to exploit the fact that only a small fraction of the locations in V is typically relevant for the SPR of a user and that unknown ratings are not missing at random. In our problem users are always asked to rate the most suitable location of a scenario. Therefore, known ratings tend to be biased towards more positive values, while unrated locations are likely to have a low suitability for a user w.r.t. an SPR. A matrix factorization approach that takes such considerations into account has been suggested by Reference [17]. Traditionally, the rating matrix $\mathcal{R}$ is factorized by solving the optimization problem

$$\min_{\xi, \nu} \sum_{i,j \mid w_{i,j} \in \mathcal{R}} E(w_{ij}, \xi_i \nu_j^{\mathrm{T}}) + \rho(\xi, \nu), \tag{14}$$

where E is a loss function for measuring the error between the actual and the predicted ratings, and ρ is a regularization term. In Reference [17], this minimization problem is expanded by adding a bias term for unknown ratings towards a certain value $\hat{w}$, i.e.,

$$\min_{\xi, \nu} \sum_{i,j \mid w_{i,j} \in \mathcal{R}} E(w_{i,j}, \xi_i \nu_j^{\mathrm{T}}) + \alpha \sum_{i,j \mid w_{i,j} \notin \mathcal{R}} E(\hat{w}, \xi_i \nu_j^{\mathrm{T}}) + \rho(\xi, \nu). \tag{15}$$

Parameter α controls the impact of this new term in the optimization. The authors show for selected loss functions how this new optimization problem can be solved in the same time complexity as the traditional optimization problem.

In order to apply matrix factorization for approximating suitability values $w_{r,v}$ in our case, we start from the sparsely filled matrix $W = (w_{r,v})_{(r,v) \in K}$ containing all so far known suitability values. By factorizing W along the r dimension and the v dimension on the basis of a feature set $F = \{1, \ldots, \phi\}$, we obtain an SPR/feature matrix $\xi = (\xi_{r,i})_{r \in R, i \in F}$ with $\xi_{r,i} \in \mathbb{R}$ and a location/feature matrix $\nu = (\nu_{v,i})_{v \in V, i \in F}$ with $\nu_{v,i} \in \mathbb{R}$. Feature vectors ξ_r describe the SPR r in terms of abstract features, while feature vector ν_v reflect the characteristics of locations v. In general, it is expected that SPRs with similar needs will have similar feature vectors in ξ, and locations with similar suitability characteristics will have similar feature vectors in ν. The number of features ϕ is hereby a parameter that is chosen, e.g., in dependence of an estimation of the overall number of different service point requirements, and we assume it is considerably smaller than the overall number of SPRs, as well as the number of locations n. As unknown suitability values are more likely zero than being greater than zero, we set the bias target $\hat{w} = 0$.

Having obtained matrices ξ and ν, an unknown value of W is approximated by the dot product of the respective feature vectors rounded to the nearest of the five discrete suitability values we defined, i.e.,

$$g_\Theta(r, v) = \lfloor 4 \cdot \xi_r \nu_v^{\mathrm{T}} + 0.5 \rfloor / 4. \tag{16}$$

The trainable parameters of g_Θ are, therefore, $\Theta = (\xi, \nu)$.

Our loss function for the matrix factorization is

$$\min \sum_{(r,v) \in K} \left(w_{r,v} - \xi_r \nu_v^{\mathrm{T}} \right)^2 + \alpha \sum_{(r,v) \notin K} (\xi_r \nu_v^{\mathrm{T}})^2 + \lambda (\|\xi_r\|^2 + \|\nu_v\|^2), \tag{17}$$

and randomized block coordinate descent [47] is used to minimize it.

4.5. Optimization Component

Recall that the OC is performed in each major iteration of the framework and makes use of the current surrogate objective function $\tilde{f}_\Theta$ provided by the EC, whose weights Θ do not change during each individual call of the OC. The OC is, thus, supposed to return an optimal or close-to-optimal solution to our problem w.r.t. the current surrogate objective function.

In our implementation of the OC, we apply a general purpose MILP solver to the MILP formulation already presented in Section 3, Equations (3)–(10); however, suitability values are approximated by the surrogate suitability function $\tilde{w}_\Theta$. Note that, for improved scalability, a metaheuristic approach might also be used as optimization core as it is not necessary to find an optimal solution in each iteration. However, due to the complexity of the objective function of the GSPDP, such a metaheuristic requires significant effort to be able to scale well to larger instances. As the development of such a heuristic would go beyond the scope of this paper, we, therefore, leave this task for future work.

5. Test Cases

The following describes how test instances for evaluating COA have been generated. Instead of testing with real users, user interaction is simulated in an idealized manner in certain test cases in order to analyze the strengths and weaknesses of the framework with a focus on the algorithmic aspects. The considered test instances are of three groups. The first two groups are purely artificial test instances inspired by the location planning of stations for electric vehicle charging, denoted by EVC, and, for (station-based) car sharing systems, denoted by CSS, respectively. While, in group EVC, each use case has one SPR, there are always two SPRs per use case in CSS. The third group of test instances is designed similarly to group CSS, also addressing the car sharing scenario, but the instances are generated from real-world taxi trip data of Manhattan; this group is, therefore, called MAN. Note that COA intentionally does not make use of geographic information in any of its components. Therefore, modeling preferences of users for our instances in dependence of the proximity to service point locations does not provide COA any advantage for finding an optimized solution.

All of our benchmark instances are available online at https://www.ac.tuwien.ac.at/research/problem-instances/#spdp (accessed on 10 June 2021).

5.1. Artificial Test Instance Groups EVC and CSS

Test instances from the groups EVC and CSS are generated with the same approach. The n possible locations for service stations are randomly distributed in the Euclidean plane with coordinates $\mathrm{coord}(v)$, $v \in V$, chosen uniformly from the grid $\{0, \ldots, L-1\}^2$, with $L = \lceil 10\sqrt{n} \rceil$. The fixed costs z_v^{fix}, as well as the variable costs z_v^{var} for setting up a service station at each location $v \in V$, are uniformly chosen at random from $\{50, \ldots, 100\}$. The budget is assumed to be $B = \lceil 7.5 \cdot n \rceil$ so that roughly 10% of the stations with average costs can be expected to be opened.

The number of use cases for each user $u \in U$ is chosen randomly according to a shifted Poisson distribution with offset one, expected value three, and a maximum value of five, i.e., the number of use cases never exceeds five. Each of these use cases $c \in C_u$ is associated with an individual demand $D_{u,c}$ randomly chosen from {5,…,50} and, depending on the benchmark group, with one (EVC) or two (CSS) SPRs.

Each SPR $r \in R_{u,c}$ of a use case c also is associated with a particular geographic location $q_r \in \{0, \ldots, L-1\}^2$. In order to model similarities in the users' SPRs, these locations are selected in the following correlated way. First, ten *attraction points* A with uniform random coordinates are selected from $\{0, \ldots, L-1\}^2$. Then, each use case location is derived by randomly choosing one of these attraction points $(a_x, a_y) \in A$ and adding a small individual offset to the coordinates, i.e.,

$$q_r = (\lfloor \mathcal{N}(a_x, \sigma_{\mathrm{v}}) \rfloor, \lfloor \mathcal{N}(a_y, \sigma_{\mathrm{v}}) \rfloor), \tag{18}$$

where $\mathcal{N}(\cdot, \cdot)$ denotes a random value sampled from a normal distribution with the respectively given mean value and standard deviation σ_{v}. If obtained coordinates are not in $\{0, \ldots, L-1\}^2$, a new attraction point is chosen, and the deviation is re-sampled.

A service point location $v \in V$ receives a rating w.r.t. an SPR r, according to a sigmoidal decay function applied to the Euclidean distance, and is also perturbed by a Gaussian noise with a standard deviation of σ_{r}:

$$w'_{r,v} = \mathcal{N}\left(\frac{1}{1 + 6e^{0.5||q_r - \mathrm{coord}(v)|| - 6}}, \sigma_{\mathrm{r}} \right). \tag{19}$$

The parameters of the sigmoid function are chosen so that $w'_{r,v}$ decreases as the distance between v and q_r increases and becomes approximately zero at a distance larger than twelve. Additionally, as motivated in Section 3, we discretize the rating $w'_{r,v}$ by rounding to the closest value in $\{0, 0.25, 0.5, 0.75, 1\}$, obtaining $w_{r,v}$. Hence, $w_{r,v} = \lfloor 4 \cdot \min(1, \max(0, w'_{r,v})) + 0.5 \rfloor / 4$.

Six sets of 30 benchmark instances were generated for EVC, as well as CSS. As detailed in Table 1, these sets consider $n \in \{100, 200, 300\}$ potential service point locations in combination with different numbers of users and two different settings for the standard deviations of the Gaussian perturbations σ_v and σ_r. Note that parameters σ_v and σ_r control how similar suitability values for locations of SPRs generated from the same attraction point are. The larger σ_v and σ_r, the more different are the preferences of the users. Hence, we have chosen two different settings for σ_v and σ_r to show how COA behaves under conditions in which preferences of users towards locations for service points have a higher and lower similarity, respectively. Values for σ_v and σ_r have been determined experimentally in preliminary tests. In the following, we will denote instance sets primarily by the pair (n, m).

Table 1. Main parameters of the EVC and CSS instance sets of groups EVC and CSS. Each row represents a set of 30 instances.

(n, m)	σ_v	σ_r
(100, 500)	3.0	0.03
(100, 1000)	5.0	0.15
(200, 1000)	3.0	0.03
(200, 2000)	5.0	0.15
(300, 1500)	3.0	0.03
(300, 3000)	5.0	0.15

5.2. Manhattan Test Instances

Next to the above described purely artificial benchmark instances, we also derive benchmark instances from real-world yellow taxi trip data of Manhattan. As in CSS, MAN instances have two SPRs per use case. The underlying street network G of the instances corresponds to the street network graph of Manhattan provided by the Julia package LightOSM (https://github.com/DeloitteDigitalAPAC/LightOSM.jl, accessed on 10 June 2021). Taxi trips have been extracted from the 2016 Yellow Taxi Trip Data (https://data.cityofnewyork.us/Transportation/2016-Yellow-Taxi-Trip-Data/k67s-dv2t, accessed on 10 June 2021). The taxi data set was first preprocessed by removing all trips with invalid data and trips made on a weekend. Furthermore, we have also removed all trips which do not start, as well as end, in Manhattan. For taxi trips within the months of January to July, geographic pickup and drop-off coordinates of customers are recorded in the data set. Each of these coordinates has been extracted and mapped to the geographically closest vertex in G, resulting in a list of pairs of vertices $Q \subseteq V(G) \times V(G)$. Next, as the similarity of users in our instances depends on their geographic proximity, we have reduced Q by considering only the ten taxi zones with the highest total number of pickups and drop-offs of customers, resulting in a total of approximately two million taxi trips. Geographic information of the taxi zones of Manhattan has been obtained from https://data.cityofnewyork.us/Transportation/NYC-Taxi-Zones/d3c5-ddgc (accessed on 10 June 2021). The left side of Figure 4 provides a visualization of the selected taxi zones.

The set of potential service point locations V has been chosen randomly from vertices of G that are located in the considered taxi zones. The fixed costs z_v^{fix}, as well as the variable costs z_v^{var}, for setting up a service station at each location $v \in V$, are uniformly chosen at random from $\{50, \dots, 100\}$.

The number of use cases for each user $u \in U$ is again chosen randomly according to a shifted Poisson distribution with offset one, expected value three, and a maximum value of five. Each of these use cases $c \in C_u$ is associated with an individual demand $D_{u,c}$ randomly chosen from $\{5, \dots, 50\}$ and the two SPRs representing the origin and destination of a trip chosen from Q uniformly at random. A rating for an SPR r is calculated for each $v \in V$ via the sigmoidal decay function

$$w'_{r,v} = \frac{1}{1 + 10e^{0.01\text{sp}(r,v)} - 6}, \tag{20}$$

where $\text{sp}(r, v)$ refers to the length of the shortest path between location v and the SPR r in the street network graph G. The parameters of this function have been chosen in such a way that service point locations within a distance of approximately 600 meters to r are relevant for the SPR. Finally, the discretized suitability value $w_{r,v}$ is again obtained by $w_{r,v} = \lfloor 4 \cdot \min(1, \max(0, w'_{r,v})) + 0.5 \rfloor / 4$. The right side of Figure 4 shows the distribution of SPR locations, as well as potential service point locations, as an example instance.

The MAN benchmark group consists of 30 instances in total with each instance having 100 potential service point locations and 2000 users. Additionally, each instance will be evaluated with different budget levels $b\,[\%] \in \{30, 50, 70\}$ such that about b percent of the stations considering average costs can be opened, i.e, the actual budget for each instance is calculated as $B = \lceil b \cdot 0.75 \cdot n \rceil$.

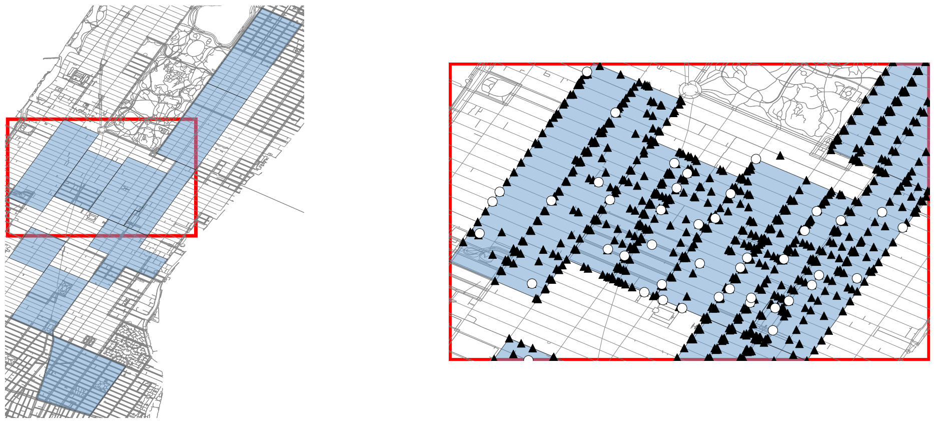

Figure 4. Left: The considered ten taxi zones of Manhattan with the highest number of pickups and drop-offs. **Right**: Exemplary distribution of SPR locations (black triangles) and potential service point locations (white points).

6. Results

The COA framework including the FC were implemented in Python 3.8. For the matrix factorization of the EC, we adapted the C++ implementation of Reference [17] provided on Github (https://github.com/rdevooght/MF-with-prior-and-updates, accessed on 10 June 2021). The parameterization of the COA components have been determined through preliminary tests on an independent set of instances. For all test runs, the weighting for unknown suitability values α of the matrix factorization has been set to one. Moreover, the number of features considered in the matrix factorization is set to ten for all test runs. The parameters ς^V and ς^* for controlling the number of scenarios generated according to each strategy in the FC have been set to 0.5 and 0.1, respectively. Finally, Gurobi 9.1 (https://www.gurobi.com/, accessed on 10 June 2021) was used to solve the MILP models in the OC. In each COA iteration, a time limit of ten minutes has been set for solving the MILP. If the MILP was not solved to optimality within this time limit, the best found solution was used. All test runs have been executed on an Intel Xeon E5-2640 v4 2.40 GHz machine in single-threaded mode with a global time limit of four hours per run. Note, however, that all runs terminated within this time limit once all relevant locations had been discovered. Since, in contrast to COA, we have full knowledge of our test instances, we are also able to calculate optimal reference solutions for each instance. Hence, by $f(x_{\text{opt}})$, we denote the objective value of a respective optimal solution x_{opt}.

To characterize the amount of user interaction performed by COA, we consider the total number of scenarios evaluated by a user $u \in U$ in relation to the upper bound of required interactions I_u^{UB}, cf. Section 4.3. Let I_u be the number of user interactions of user $u \in U$ performed within COA to generate some solution. Then, $I = 100\% \cdot (\sum_{u \in U} I_u / I_u^{\text{UB}})/m$ refers to the relative average number of performed user interactions relative to I_u^{UB} over all users. Note that since scenarios are presented only to a fraction of users in every iteration, the average number of user interactions at each iteration of COA varies even for instances within the same benchmark group. Hence, in order to reasonably study results for each of our benchmark groups, we aggregate respective results to our instances at various *interaction levels* ψ by selecting for each instance the COA iteration at which I is largest but does not exceed ψ. Note that, for some instances, smaller levels of ψ are already exceeded in the first iteration of COA. Hence, in the following, we only consider interaction levels for an instance group for which results to all corresponding instances exist.

Note further that the user interaction levels can also be interpreted as the average information known about a user. This interpretation allows us to draw a direct comparison to traditional approaches for distributing service points in which information about the demands of the users is determined in advance. Each result at a certain interaction level can also be interpreted as the result of such a traditional approach with a certain level of knowledge about the users. However, to the best of our knowledge, there exists no data about suitability values in other work. Additionally, for a fair comparison between COA and other approaches from literature, one would have to also take into account the costs required for obtaining said information about the users. Therefore, comparing COA to other approaches from literature seems to be not possible without an extensive study on suitability values of users or a complex simulation of users based on various assumptions that can heavily influence the outcome of such a comparison.

First, we want to show how the quality of incumbent solutions develops as the number of user interactions increases during a COA run. For this purpose, we calculate the optimality gap for a solution x obtained from COA as gap $= 100\% \cdot (f(x_{\text{opt}}) - f(x))/f(x_{\text{opt}})$. Figure 5 shows the average optimality gaps of solutions to each of our benchmark sets against the interaction levels ψ. The results are grouped by σ_v and σ_r to additionally compare instances groups with similar user behavior.

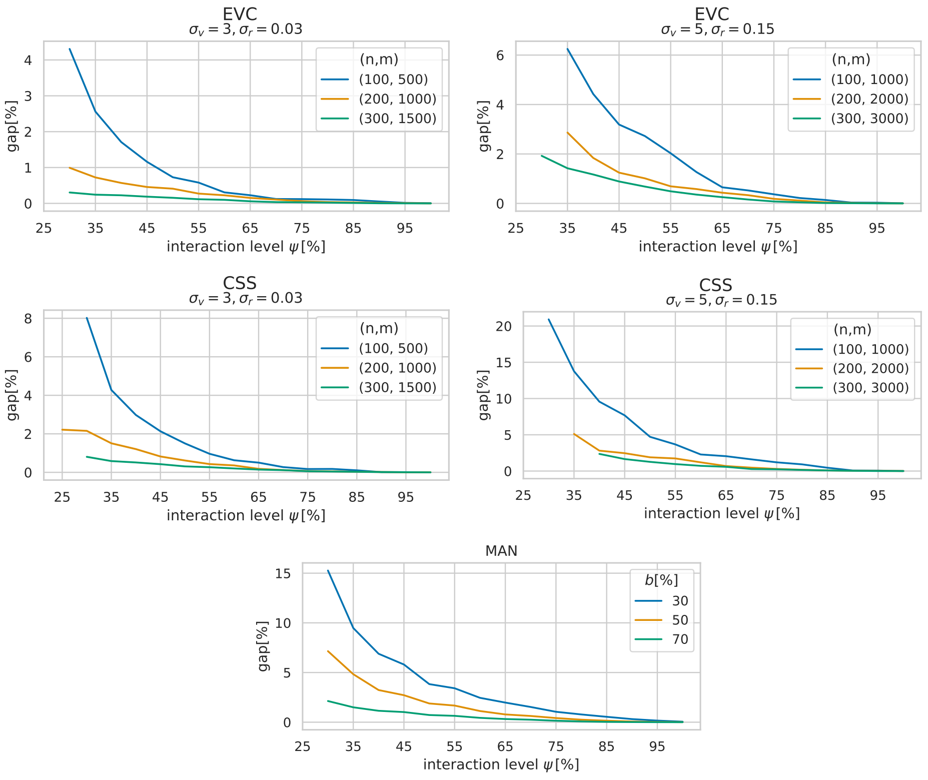

Figure 5. Development of average optimality gaps with an increasing number of user interactions for each benchmark instance set.

Recall that the number of attraction points is the same for all EVC and CSS instances. Therefore, for instances with a higher number of users, it is generally easier to find better solutions as there are more users that prefer the same locations. The plots show that, in all cases, solutions generally improve quickly with an increasing interaction level, and close to optimal solutions can be obtained well before identifying all the users' relevant locations. Specifically, at a user interaction level of 50%, the solutions generated by COA feature optimality gaps of 1.45% on average. An exception of this observation are the MAN instances with $b = 30\%$. For these, generated solutions do not reach an optimality gap below 1% before $\psi = 80\%$, on average. Moreover, the figure also shows that the solutions to MAN instances generally converge notably slower than the solutions to EVC and CSS instances. This behavior is likely caused by the weaker correlation of user preferences in the MAN instances and the way that the FC generates scenarios presented to the users, specifically the aspect that locations important to the individual users are tried to be identified first. Locations important to individual users might not necessarily be the best locations to add to a solution, especially if there is a lower number of users with similar preferences. Consequently, the strategies based on which scenarios are generated may still have some room for improvement for such cases. Instead of primarily identifying locations important to users, targeting locations in relation to the current best solution with a higher emphasis might be a more expedient approach here.

Note that an increased number of user interactions does not only imply a larger trainings set for the surrogate function but also results in better upper bounds $w_{r,v}^{\mathrm{UB}}$ for locations $v \in V$ w.r.t. to an SPR $r \in R$. Therefore, to gain a better understanding of how much the surrogate function actually contributes to finding an optimized solution, we study what happens when the learning surrogate suitability function $\tilde{w}_\Theta$ is replaced by the naive function with no learning capabilities

$$\tilde{w}_{\mathrm{bl}}(r,v) = \begin{cases} w_{r,v} & \text{if } (r,v) \in K \\ 0 & \text{else.} \end{cases} \tag{21}$$

In the following, we refer to our original COA implementation with the surrogate function $\tilde{w}_\Theta$ as COA$[\tilde{w}_\Theta]$ and denote the implementation with the naive function $\tilde{w}_{\mathrm{bl}}$ as COA$[\tilde{w}_{\mathrm{bl}}]$. A comparison between COA$[\tilde{w}_\Theta]$ and COA$[\tilde{w}_{\mathrm{bl}}]$ is shown in Table 2. Each table cell shows the average optimality gaps of solutions to the respective instance group at the specified interaction levels ψ. The better results among COA$[\tilde{w}_\Theta]$ and COA$[\tilde{w}_{\mathrm{bl}}]$ are printed bold. Additionally, as the standard deviations w.r.t. the optimality gaps are quite large, shown in Figure 6, we have also applied a one-sided Wilcoxon signed-rank test to determine for each group, whether the difference in optimality gaps is significant or not. Instance groups for which the Wilcoxon test has assessed at a 95% confidence interval that either COA$[\tilde{w}_\Theta]$ or COA$[\tilde{w}_{\mathrm{bl}}]$ has produced better optimality gaps are marked with an asterisk.

The table shows that, especially for the CSS and MAN instances, COA$[\tilde{w}_\Theta]$ generates significantly better results at almost all interaction levels ψ than COA$[\tilde{w}_{\mathrm{bl}}]$. While, for the EVC instances, the average optimality gaps w.r.t COA$[\tilde{w}_\Theta]$ are lower than the average optimality gaps w.r.t. COA$[\tilde{w}_{\mathrm{bl}}]$, there are instance groups for which no significant difference between the optimality gaps can be determined. However, there is no instance group for which COA$[\tilde{w}_{\mathrm{bl}}]$ produced significantly better results than COA with $\tilde{w}_\Theta$ over all user interaction thresholds. It can be observed that, at very low levels of user interaction, COA$[\tilde{w}_\Theta]$ and COA$[\tilde{w}_{\mathrm{bl}}]$ seem to be equally strong. However, as the amount of collected of user feedback increases, COA$[\tilde{w}_\Theta]$ quite quickly outperforms COA$[\tilde{w}_{\mathrm{bl}}]$.

Figure 6 gives a visual comparison between COA$[\tilde{w}_\Theta]$ and COA$[\tilde{w}_{\mathrm{bl}}]$ for selected instance groups and not only shows average optimality gaps but also respective standard deviations around the mean values as shaded areas. The figure confirms that the average gaps produced by COA$[\tilde{w}_\Theta]$ are generally lower than those of COA$[\tilde{w}_{\mathrm{bl}}]$ but also shows that the standard deviations are quite large in general for both approaches. However, as COA progresses and the quality of the solutions improves, the standard deviations decrease, as well.

To further investigate the learning capabilities of the surrogate function, we now look at the mean squared error (MSE) of $\tilde{w}_\Theta$ with respect to the known exact values w. The left plots in Figure 7 show the development of this MSE calculated over all suitability values that are not known yet by COA for all instance groups. It can be seen that the MSE is generally small and approaches zero rather quickly. The reason for such small values can be found in the matrix factorization model used in the EC, which adds a bias for unknown suitability values towards zero as users typically only have a small number of locations with positive suitability values for each of their SPRs. Consequently, the MSE is distorted by the large number suitability values that are zero.

Table 2. Average optimality gaps obtained by COA using the surrogate suitability functions with ($\tilde{w}_{\Theta}$) and without learning capabilities ($\tilde{w}_{\text{bl}}$) at different interaction levels.

EVC												
(n, m)	**(100, 500)**		**(100, 1000)**		**(200, 1000)**		**(200, 2000)**		**(300, 1500)**		**(300, 3000)**	
ψ	**COA[$\tilde{w}_{\Theta}$]**	**COA[$\tilde{w}_{\text{bl}}$]**	**COA[$\tilde{w}_{\Theta}$]**	**COA[$\tilde{w}_{\text{bl}}$]**	**COA[$\tilde{w}_{\Theta}$]**	**COA[$\tilde{w}_{\text{bl}}$]**	**COA[$\tilde{w}_{\Theta}$]**	**COA[$\tilde{w}_{\text{bl}}$]**	**COA[$\tilde{w}_{\Theta}$]**	**COA[$\tilde{w}_{\text{bl}}$]**	**COA[$\tilde{w}_{\Theta}$]**	**COA[$\tilde{w}_{\text{bl}}$]**
30%	4.31	**4.03**	-	-	**0.99**	0.99	-	-	**0.30**	0.32	**1.93**	1.94
40%	**1.71**	2.00	**4.42**	4.95	**0.57**	0.62	**1.84**	2.06	**0.23**	0.23	**1.17** *	1.31
50%	**0.73** *	1.11	**2.72**	3.03	**0.41**	0.46	**1.01** *	1.44	**0.16**	0.16	**0.68** *	0.86
60%	**0.31** *	0.69	**1.27** *	1.66	**0.23** *	0.29	**0.58** *	0.91	0.10	**0.09**	**0.36** *	0.49
70%	**0.12** *	0.42	**0.53** *	0.90	**0.11**	0.14	**0.33** *	0.50	**0.04** *	0.06	**0.16** *	0.30
80%	**0.11** *	0.18	**0.22** *	0.53	**0.04** *	0.08	**0.11** *	0.21	**0.02**	0.04	**0.05** *	0.10
90%	**0.05**	0.11	**0.03** *	0.13	**0.01**	0.03	**0.02** *	0.06	**0.00** *	0.01	**0.01** *	0.02
CSS												
(n, m)	**(100, 500)**		**(100, 1000)**		**(200, 1000)**		**(200, 2000)**		**(300, 1500)**		**(300, 3000)**	
ψ	**COA[$\tilde{w}_{\Theta}$]**	**COA[$\tilde{w}_{\text{bl}}$]**	**COA[$\tilde{w}_{\Theta}$]**	**COA[$\tilde{w}_{\text{bl}}$]**	**COA[$\tilde{w}_{\Theta}$]**	**COA[$\tilde{w}_{\text{bl}}$]**	**COA[$\tilde{w}_{\Theta}$]**	**COA[$\tilde{w}_{\text{bl}}$]**	**COA[$\tilde{w}_{\Theta}$]**	**COA[$\tilde{w}_{\text{bl}}$]**	**COA[$\tilde{w}_{\Theta}$]**	**COA[$\tilde{w}_{\text{bl}}$]**
30%	8.02	**7.60**	20.92	**19.11** *	**2.15**	2.31	-	-	**0.81**	0.82	-	-
40%	**2.98** *	4.28	9.57	**9.52**	**1.21** *	1.63	**2.81** *	3.78	**0.51**	0.52	**2.36** *	2.64
50%	**1.50** *	2.41	4.72	**4.41**	**0.62** *	0.99	**1.90** *	2.77	**0.31** *	0.38	**1.27** *	1.80
60%	**0.63** *	1.51	**2.29** *	3.76	**0.36** *	0.63	**1.20** *	1.83	**0.20** *	0.26	**0.72** *	1.15
70%	**0.27** *	0.51	**1.61** *	2.36	**0.12** *	0.37	**0.47** *	1.09	**0.11** *	0.15	**0.28** *	0.59
80%	**0.18** *	0.43	**0.92** *	1.44	**0.05** *	0.12	**0.18** *	0.46	**0.04** *	0.07	**0.15** *	0.29
90%	**0.01** *	0.14	**0.08** *	0.65	**0.02** *	0.06	**0.05** *	0.15	**0.01**	0.02	**0.03** *	0.07
MAN												
b	**30%**		**50%**		**70%**							
ψ	COA[$\tilde{w}_{\Theta}$]	COA[$\tilde{w}_{\text{bl}}$]	**COA[$\tilde{w}_{\Theta}$]**	**COA[$\tilde{w}_{\text{bl}}$]**	**COA[$\tilde{w}_{\Theta}$]**	**COA[$\tilde{w}_{\text{bl}}$]**						
30%	15.61	**14.93**	**7.02**	7.28	2.13	**2.13**						
40%	**6.25** *	7.53	**3.01** *	3.46	**1.12**	1.18						
50%	**3.34** *	4.33	**1.63** *	2.14	**0.67** *	0.77						
60%	**2.10** *	2.80	**0.93** *	1.32	**0.39** *	0.48						
70%	**1.20** *	1.86	**0.46** *	0.80	**0.22** *	0.28						
80%	**0.52** *	1.04	**0.18** *	0.33	**0.07** *	0.09						
90%	**0.24** *	0.38	**0.04** *	0.10	**0.01**	0.01						

Therefore, the plots on the right side of Figure 7 show average mean squared errors calculated only over all *positive* suitability values that are not known yet; we denote this error by MSE+. This measure gives a clearer picture on how the surrogate function continuously improves in all cases with an increasing amount of gained knowledge. At the start of the algorithm the MSEs of the benchmark groups are between 0.2 and 0.3 on average and go towards zero almost linearly with the interaction level. Note that neither the size of an instance nor the given budget seem to have a significant impact on the size of the errors. Additionally, the figure also highlights how the instance parameters σ_v and σ_r impact the similarity of user preferences as the MSEs for instances with $\sigma_v = 5$ and $\sigma_r = 0.15$ are generally larger than the MSEs for instances with $\sigma_v = 3$ and $\sigma_r = 0.03$.

Finally, in Table 3, we also compare COA$[\tilde{w}_{\Theta}]$ to the COA implementation using the surrogate function $\tilde{w}'_{\Theta}$ introduced in Reference [16], henceforth referred to as COA$[\tilde{w}'_{\Theta}]$. While $\tilde{w}'_{\Theta}$ is also based on a matrix factorization model, this model does not take into account that user data is not missing at random.

Each table cell shows the average optimality gaps of solutions to the respective instance set at the specified interaction level ψ. The better results among COA$[\tilde{w}_{\Theta}]$ and COA$[\tilde{w}'_{\Theta}]$ are printed bold. A one-sided Wilcoxon signed-rank test was used to determine for each instance set whether the difference in optimality gaps is significant or not. Entries for which the Wilcoxon test has assessed at a 95% confidence level that either COA$[\tilde{w}_{\Theta}]$ or COA$[\tilde{w}'_{\Theta}]$ has produced better optimality gaps are marked with an asterisk. It can be observed that, for lower levels of user interaction, solutions generated by COA$[\tilde{w}'_{\Theta}]$ exhibit extremely high optimality gaps. However, the higher the number of user interactions, the more COA$[\tilde{w}'_{\Theta}]$ can catch up with COA$[\tilde{w}_{\Theta}]$. However, most of the time, COA$[\tilde{w}_{\Theta}]$ still dominates COA$[\tilde{w}'_{\Theta}]$. Summarizing, it can be concluded that the new matrix factorization model is a significant improvement over our previous model $\tilde{w}'_{\Theta}$, resulting in COA$[\tilde{w}_{\Theta}]$ generating better solutions with fewer user interactions than COA$[\tilde{w}'_{\Theta}]$ most of the time.

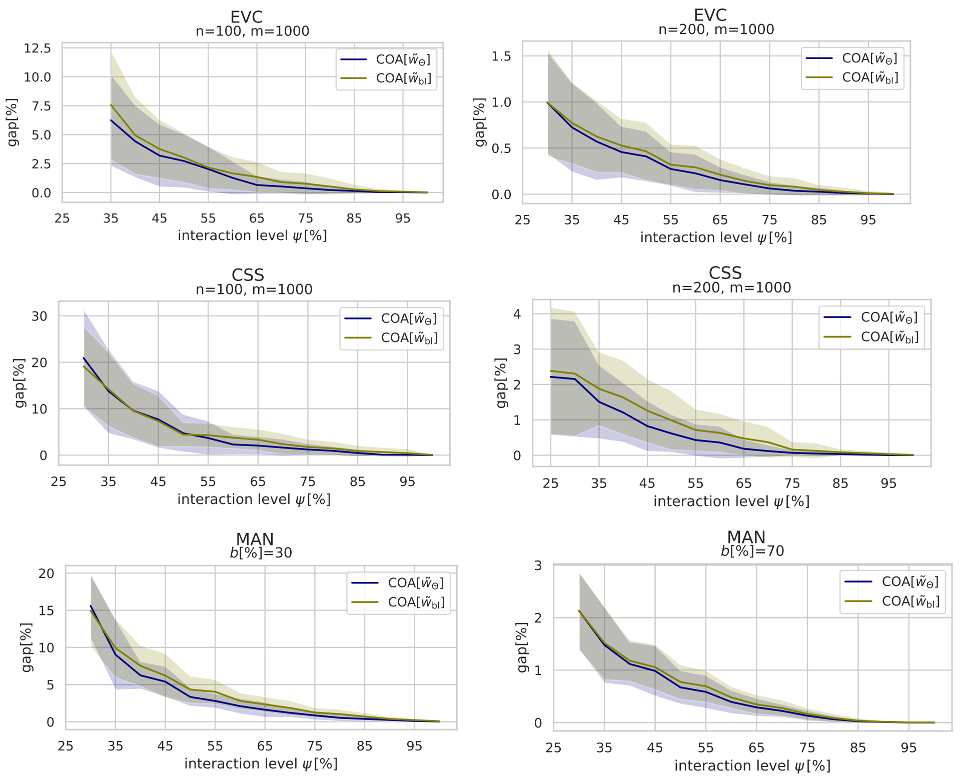

Figure 6. Average optimality gaps with standard deviations as shaded areas obtained by COA using the surrogate suitability functions with learning capabilities ($\tilde{w}_\Theta$) and without ($\tilde{w}_{bl}$) plotted over the interaction level.

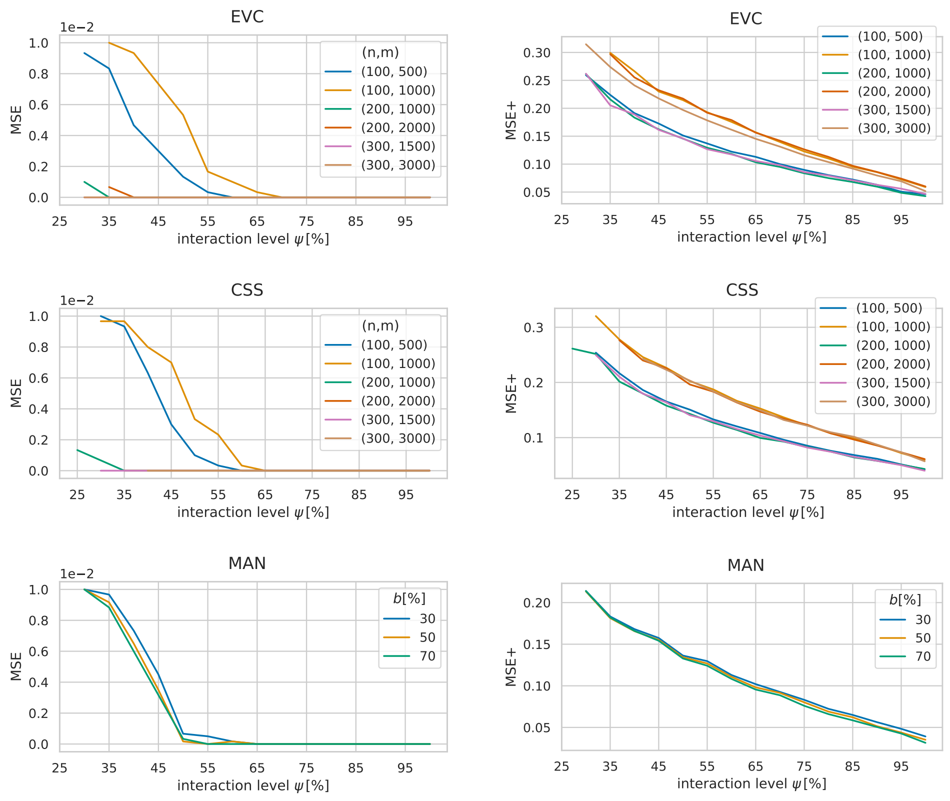

Figure 7. Development of average MSEs of $\tilde{w}_\Theta$ over all so far unknown suitability values (left) and all so far unknown *positive* suitability values (right) for all benchmark sets.

Table 3. Average optimality gaps of solution from COA[$\tilde{w}_\Theta$] and COA[$\tilde{w}'_\Theta$], where the latter utilizes the former surrogate function $\tilde{w}'_\Theta$ from Reference [16].

EVC

(n, m)	**(100, 500)**		**(100, 1000)**		**(200, 1000)**		**(200, 2000)**		**(300, 1500)**		**(300, 3000)**	
ψ	**COA[$\tilde{w}'_\Theta$]**	**COA[$\tilde{w}_\Theta$]**	**COA[$\tilde{w}'_\Theta$]**	**COA[$\tilde{w}_\Theta$]**	**COA[$\tilde{w}'_\Theta$]**	**COA[$\tilde{w}_\Theta$]**	**COA[$\tilde{w}'_\Theta$]**	**COA[$\tilde{w}_\Theta$]**	**COA[$\tilde{w}'_\Theta$]**	**COA[$\tilde{w}_\Theta$]**	**COA[$\tilde{w}'_\Theta$]**	**COA[$\tilde{w}_\Theta$]**
30%	81.89	**4.31** *	-	-	85.20	**0.99** *	-	-	82.14	**0.30** *	95.53	**1.93** *
40%	16.07	**1.71** *	34.82	**4.42** *	3.22	**0.57**	32.79	**1.84** *	9.08	**0.23**	23.21	**1.17** *
50%	1.95	**0.73** *	3.91	**2.72** *	**0.35** *	0.41	1.51	**1.01** *	**0.13** *	0.16	0.87	**0.68** *
60%	0.62	**0.31** *	2.20	**1.27** *	0.24	**0.23**	0.76	**0.58** *	**0.09**	0.10	0.48	**0.36** *
70%	0.46	**0.12** *	0.70	**0.52**	0.13	**0.11**	0.45	**0.33** *	0.04	**0.04**	0.24	**0.16** *
80%	0.22	**0.11** *	0.31	**0.22**	0.06	**0.04** *	0.17	**0.11** *	**0.02**	0.02	0.11	**0.05** *
90%	0.09	**0.05**	0.09	**0.03** *	0.02	**0.01**	0.05	**0.02** *	0.01	**0.00**	0.02	**0.01** *

CSS

(n, m)	**(100, 500)**		**(100, 1000)**		**(200, 1000)**		**(200, 2000)**		**(300, 1500)**		**(300, 3000)**	
ψ	**COA[$\tilde{w}'_\Theta$]**	**COA[$\tilde{w}_\Theta$]**	**COA[$\tilde{w}'_\Theta$]**	**COA[$\tilde{w}_\Theta$]**	**COA[$\tilde{w}'_\Theta$]**	**COA[$\tilde{w}_\Theta$]**	**COA[$\tilde{w}'_\Theta$]**	**COA[$\tilde{w}_\Theta$]**	**COA[$\tilde{w}'_\Theta$]**	**COA[$\tilde{w}_\Theta$]**	**COA[$\tilde{w}'_\Theta$]**	**COA[$\tilde{w}_\Theta$]**
30%	80.12	**8.02** *	98.68	**20.92** *	86.58	**2.15** *	-	-	69.78	**0.81** *	-	-
40%	13.80	**2.98** *	24.73	**9.57** *	1.38	**1.21**	13.93	**2.81** *	3.84	**0.51**	6.17	**2.36** *
50%	2.34	**1.50**	7.31	**4.72** *	0.77	**0.62**	2.66	**1.90** *	0.33	**0.31**	4.73	**1.27**
60%	1.83	**0.63** *	2.53	**2.29**	0.49	**0.36** *	1.56	**1.20** *	0.22	**0.20**	0.83	**0.72**
70%	0.66	**0.27** *	1.91	**1.61**	0.21	**0.12**	0.74	**0.47** *	0.12	**0.11**	0.59	**0.28** *
80%	**0.12**	0.18	0.99	**0.92**	0.10	**0.05** *	0.42	**0.18** *	**0.04**	0.04	0.23	**0.15** *
90%	0.08	**0.01** *	0.45	**0.08** *	0.04	**0.02**	0.20	**0.05** *	0.02	**0.01**	0.13	**0.03** *

MAN

b	**30%**		**50%**		**70%**	
ψ	**COA[$\tilde{w}'_\Theta$]**	**COA[$\tilde{w}_\Theta$]**	**COA[$\tilde{w}'_\Theta$]**	**COA[$\tilde{w}_\Theta$]**	**COA[$\tilde{w}'_\Theta$]**	**COA[$\tilde{w}_\Theta$]**
30%	99.78	**15.61** *	99.83	**7.02** *	99.83	**2.13** *
40%	13.84	**6.25** *	3.97	**3.01** *	1.15	**1.12**
50%	8.46	**3.34** *	1.93	**1.63** *	**0.63**	0.67
60%	3.18	**2.10** *	**0.89**	0.93	**0.35**	0.39
70%	1.86	**1.20** *	0.64	**0.46** *	0.26	**0.22**
80%	1.15	**0.53** *	0.34	**0.18** *	0.12	**0.07** *
90%	0.47	**0.24** *	0.14	**0.04** *	0.03	**0.01** *

7. Conclusions and Future Work

In this paper, the previously introduced Cooperative Optimization Algorithm was generalized to be applicable to more application scenarios and to larger instances with hundreds of potential service station locations and thousands of users. New application scenarios include, in particular, those in which the fulfillment of a single demand depends on more than one suitably located service points as it is the case in station-based bike and care sharing. Results on artificial and real world inspired instances show how the solution quality improves as the amount of user feedback increases and that a near optimal solution is reached for most instances with a reasonably low amount of user interactions. To characterize the amount of user interaction performed by COA, we have established an upper bound on the maximum number of non-redundant user interactions and introduced the notion of user interaction levels. Solutions generated by COA feature optimality gaps of 1.45% on average at an interaction level of 50%. Furthermore, we could clearly observe that the matrix factorization-based surrogate model is able to learn preferences of individual users from users with similar interests. More specifically, we made use of an advanced matrix factorization model which takes into account that user data is not missing at random. In fact, users are always asked to rate the most suitable location of a scenario, if one exists. The experimental comparison indeed confirmed the benefits of this new model over the original one, especially when the interaction level is still low. Using the new matrix factorization model, COA is able to generate better solutions with fewer known user preferences than before. In addition, note that, while we achieve our best results with the matrix factorization-based surrogate model, the results also show that COA already works reasonable even without learning user preferences.

However, there is still potential left for future improvements. In our COA implementation, the strategies by which scenarios for users are generated favor the selection of unrated locations that may be important for individual users but not necessarily for a global optimal solution. In order to quickly find a good solution by the optimization, it is important for our surrogate function to have higher accuracy for locations that have the potential to actually appear in a globally optimal solution. Otherwise, finding a near optimal solution requires a larger amount of user interactions as we have observed in our results on the Manhattan instances. In order to improve the scenario generation strategies, it seems natural to enrich the feedback component (FC) with knowledge not only from the evaluation component (EC) but also from the optimization component (OC). As the OC finds optimized solutions via a MILP, utilizing dual solution information, such as reduced costs or performing a sensitivity analysis, might be a promising direction.

It would also be interesting to further improve the scalability of COA. While a time limit of ten minutes per MILP was still sufficient for solving our benchmark instances, the OC is the main bottleneck of COA w.r.t. computation times. On the one hand, one can resort to heuristic optimization approaches in the OC. On the other hand, hierarchical clustering and multilevel refinement strategies as applied in the context of planning a bike sharing system in Reference [2] appear promising.

COA was applied for generating solutions to instances of the General Service Point Distribution Problem (GSPDP). While we have proven the GSPDP to be NP-hard, this problem is rather abstract and, from a practical perspective, still too simplistic. For a specific practical application, the problem formulation needs to be tailored appropriately. Diverse aspects, such as different configuration options of stations, capacities, or time-dependent aspects, may be needed to be considered. To a certain degree, the general framework of COA can stay the same or may need only smaller adaptions, such as in the scenario generation of the FC.

Finally, we want to emphasize that the focus of this contribution was on the algorithmic and computational aspects of COA and its components. Clearly, further challenges concern a suitable user interface and a corresponding distributed implementation of at least the FC, in which also psychological aspects of users need to be considered. Moreover, the performed experiments are based on the assumption of perfect user feedback, which does not hold in practice. The impacts of not entirely reliable evaluation results need to be studied, and robust variants of certain components of COA need to be devised.

Author Contributions: Conceptualization, T.R.; methodology, T.R., G.R.R. and T.J.; software, validation, T.J.; writing—original draft preparation, T.J. and G.R.R.; writing—review and editing, T.J., G.R.R. and T.R.; supervision, G.R.R. and T.R. All authors have read and agreed to the published version of the manuscript.

Funding: Thomas Jatschka was financially supported by the Honda Research Institute Europe. Open Access Funding by TU Wien.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: All of our benchmark instances are available online at https://www.ac.tuwien.ac.at/research/problem-instances/#spdp (accessed on 10 June 2021). One instance group was derived from the 2016 Yellow Taxi Trip Dataset: https://data.cityofnewyork.us/Transportation/2016-Yellow-Taxi-Trip-Data/k67s-dv2t (accessed on 10 June 2021).

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

References

1. Gavalas, D.; Konstantopoulos, C.; Pantziou, G. Design and Management of Vehicle-Sharing Systems: A Survey of Algorithmic Approaches. In *Smart Cities and Homes*; Obaidat, M.S., Nicopolitidi, P., Eds.; Elsevier: Amsterdam, The Netherlands, 2016; pp. 261–289.
2. Kloimüllner, C.; Raidl, G.R. Hierarchical Clustering and Multilevel Refinement for the Bike-Sharing Station Planning Problem. In *International Conference on Learning and Intelligent Optimization*; Springer: Berlin/Heidelberg, Germany, 2017; Volume 10556, pp. 150–165.
3. Xu, Y.; Shaw, S.L.; Fang, Z.; Yin, L. Estimating Potential Demand of Bicycle Trips from Mobile Phone Data—An Anchor-Point Based Approach. *ISPRS Int. J. Geo-Inf.* **2016**, *5*, 131. [CrossRef]

4. Wang, C.; Bi, J.; Sai, Q.; Yuan, Z. Analysis and Prediction of Carsharing Demand Based on Data Mining Methods. *Algorithms* **2021**, *14*, 179. [CrossRef]
5. Schmidt, M.; Zmuda-Trzebiatowski, P.; Kiciński, M.; Sawicki, P.; Lasak, K. Multiple-Criteria-Based Electric Vehicle Charging Infrastructure Design Problem. *Energies* **2021**, *14*, 3214. [CrossRef]
6. Almaghrebi, A.; Aljuheshi, F.; Rafaie, M.; James, K.; Alahmad, M. Data-Driven Charging Demand Prediction at Public Charging Stations Using Supervised Machine Learning Regression Methods. *Energies* **2020**, *13*, 4231. [CrossRef]
7. Awasthi, A.; Venkitusamy, K.; Padmanaban, S.; Selvamuthukumaran, R.; Blaabjerg, F.; Singh, A.K. Optimal planning of electric vehicle charging station at the distribution system using hybrid optimization algorithm. *Energy* **2017**, *133*, 70–78. [CrossRef]
8. Cavadas, J.; Homem, G.d.A.C.; Gouveia, J. A MIP Model for Locating Slow-Charging Stations For Electric Vehicles in Urban Areas Accounting for Driver Tours. *Transp. Res. Part E Logist. Transp. Rev.* **2015**, *75*, 188–201. [CrossRef]
9. Dong, J.; Liu, C.; Lin, Z. Charging infrastructure planning for promoting battery electric vehicles: An activity-based approach using multiday travel data. *Transp. Res. Part C Emerg. Technol.* **2014**, *38*, 44–55. [CrossRef]
10. Pagany, R.; Camargo, L.R.; Dorner, W. A review of spatial localization methodologies for the electric vehicle charging infrastructure. *Int. J. Sustain. Transp.* **2019**, *13*, 433–449. [CrossRef]
11. Molin, E.; Mokhtarian, P.; Kroesen, M. Multimodal travel groups and attitudes: A latent class cluster analysis of Dutch travelers. *Transp. Res. Part A Policy Pract.* **2016**, *83*, 14–29. [CrossRef]
12. Radzimski, A.; Dzięcielski, M. Exploring the relationship between bike-sharing and public transport in Poznań, Poland. *Transp. Res. Part A Policy Pract.* **2021**, *145*, 189–202. [CrossRef]
13. Jatschka, T.; Rodemann, T.; Raidl, G.R. A Cooperative Optimization Approach for Distributing Service Points in Mobility Applications. In *Evolutionary Computation in Combinatorial Optimization*; Liefooghe, A., Paquete, L., Eds.; Springer: Berlin/Heidelberg, Germany, 2019; Volume 11452, pp. 1–16.
14. Meignan, D.; Knust, S.; Frayret, J.M.; Pesant, G.; Gaud, N. A Review and Taxonomy of Interactive Optimization Methods in Operations Research. *ACM Trans. Interact. Intell. Syst.* **2015**, *5*, 17:1–17:43. [CrossRef]
15. Belton, V.; Branke, J.; Eskelinen, P.; Greco, S.; Molina, J.; Ruiz, F.; Słowiński, R. Interactive Multiobjective Optimization from a Learning Perspective. In *Multiobjective Optimization: Interactive and Evolutionary Approaches*; Branke, J., Deb, K., Miettinen, K., Słowiński, R., Eds.; Springer: Berlin/Heidelberg, Germany, 2008; pp. 405–433.
16. Jatschka, T.; Rodemann, T.; R. Raidl, G. Exploiting Similar Behavior of Users in a Cooperative Optimization Approach for Distributing Service Points in Mobility Applications. In *Machine Learning, Optimization, and Data Science*; Nicosia, G., Pardalos, P., Giuffrida, G., Umeton, R., Sciacca, V , Eds.; Springer: Berlin/Heidelberg, Germany, 2019; Volume 11943, pp. 738–750.
17. Devooght, R.; Kourtellis, N.; Mantrach, A. Dynamic Matrix Factorization with Priors on Unknown Values. In Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, Australi , 10–13 August 2015; pp. 189–198.
18. Laporte, G.; Nickel, S.; Saldanha-da Gama, F. *Location Science*; Springer: Berlin/Heidelberg, Germany, 2015.
19. Cornuéjols, G.; Nemhauser, G.L.; Wolsey, L.A. The Uncapacitated Facility Location Problem. In *Discrete Location Theory*; Mirchandani, P.B., Francis, R.L., Eds.; Wiley: Hoboken, NJ, USA, 1990; pp. 119–171.
20. Snyder, L.V. Facility location under uncertainty: A review. *IIE Trans.* **2006**, *38*, 547–564. [CrossRef]
21. Albareda-Sambola, M.; Fernández, E.; da Gama, F.S. The facility location problem with Bernoulli demands. *Omega* **2011**, *39*, 335–345. [CrossRef]
22. Turkeš, R.; Sörensen, K.; Cuervo, D.P. A matheuristic for the stochastic facility location problem. *J. Heuristics* **2021**, *27*, 649–694 [CrossRef]
23. Farahani, R.Z.; Hekmatfar, M. *Facility Location: Concepts, Models, Algorithms and Case Studies*; Springer: Berlin/Heidelberg, Germany, 2009.
24. Chen, T.; Kockelman, K.M.; Khan, M. The Electric Vehicle Charging Station Location Problem: A Parking-Based Assignment Method for Seattle. In Proceedings of the 92nd Annual Meeting of the Transportation Research Board, Washington, DC, USA, 13–17 January 201 .
25. Kameda, H.; Mukai, N. Optimization of Charging Station Placement by Using Taxi Probe Data for On-Demand Electrical Bus System. In *Knowledge-Based and Intelligent Information and Engineering Systems*; König, A., Dengel, A., Hinkelmann, K., Kise, K., Howlett, R.J., Jain, L.C., Eds.; Springer: Berlin/Heidelberg, Germany, 2011; pp. 606–615.
26. Boyacı, B.; Zografos, K.G.; Geroliminis, N. An optimization framework for the development of efficient one-way car-sharing systems. *Eur. J. Oper. Res.* **2015**, *240*, 718–733. [CrossRef]
27. Frade, I.; Ribeiro, A. Bike-sharing stations: A maximal covering location approach. *Transp. Res. Part A Policy Pract.* **2015**, *82*, 216–227. [CrossRef]
28. Frade, I.; Ribeiro, A.; Gonçalves, G.; Antunes, A. Optimal Location of Charging Stations for Electric Vehicles in a Neighborhood in Lisbon, Portugal. *Transp. Res. Rec. J. Transp. Res. Board* **2011**, *2252*, 91–98. [CrossRef]
29. Vogel, P.; Greiser, T.; Mattfeld, D.C. Understanding Bike-Sharing Systems using Data Mining: Exploring Activity Patterns. *Procedia Soc. Behav. Sci.* **2011**, *20*, 514–523. [CrossRef]
30. Zhou, X. Understanding Spatiotemporal Patterns of Biking Behavior by Analyzing Massive Bike Sharing Data in Chicago. *PLoS ONE* **2015**, *10*, e0137922. [CrossRef]

31. Trentini, A.; Losacco, F. Analyzing Carsharing "Public" (Scraped) Data to Study Urban Traffic Patterns. *Procedia Environ. Sci.* **2017**, *37*, 594–603. [CrossRef]
32. Ciari, F.; Schuessler, N.; Axhausen, K.W. Estimation of Carsharing Demand Using an Activity-Based Microsimulation Approach: Model Discussion and Some Results. *Int. J. Sustain. Transp.* **2013**, *7*, 70–84. [CrossRef]
33. Horni, A.; Nagel, K.; Axhausen, K.W. *Multi-Agent Transport Simulation MATSim*; Ubiquity Press: London, UK, 2016.
34. Correia, G.H.D.A.; Jorge, D.R.; Antunes, D.M. The Added Value of Accounting For Users' Flexibility and Information on the Potential of a Station-Based One-Way Car-Sharing System: An Application in Lisbon, Portugal. *J. Intell. Transp. Syst.* **2014**, *18*, 299–308. [CrossRef]
35. Llorà, X.; Sastry, K.; Goldberg, D.E.; Gupta, A.; Lakshmi, L. Combating User Fatigue in iGAs: Partial Ordering, Support Vector Machines, and Synthetic Fitness. In Proceedings of the 7th Annual Conference on Genetic and Evolutionary Computation, Washington, DC, USA , 25–29 June 2005 ; pp. 1363–1370.
36. Sun, X.; Gong, D.; Jin, Y.; Chen, S. A New Surrogate-Assisted Interactive Genetic Algorithm With Weighted Semisupervised Learning. *IEEE Trans. Cybern.* **2013**, *43*, 685–698. [PubMed]
37. Sun, X.Y.; Gong, D.; Li, S. Classification and Regression-based Surrogate Model-assisted Interactive Genetic Algorithm with Individual's Fuzzy Fitness. In Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation, Montreal, Canada, 8–12 July 2009; pp. 907–914.
38. Koziel, S.; Ciaurri, D.E.; Leifsson, L. Surrogate-based Methods. Computational Optimization, Methods and Algorithms. In *Studies in Computational Intelligence*; Springer: Berlin/Heidelberg, Germany, 2011; Volume 356, pp. 33–59.
39. Bell, R.M.; Koren, Y.; Volinsky, C. Matrix Factorization Techniques for Recommender Systems. *Computer* **2009**, *42*, 30–37.
40. Ekstrand, M.D.; Riedl, J.T.; Konstan, J.A. Collaborative Filtering Recommender Systems. *Found. Trends Hum. Comput. Interact.* **2011**, *4*, 81–173. [CrossRef]
41. Shi, L.; Rasheed, K. ASAGA: An Adaptive Surrogate-assisted Genetic Algorithm. In Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation, Atlanta, GA, USA, 12–16 July 2008 ; pp. 1049–1056.
42. Mladenović, N.; Hansen, P. Variable Neighborhood Search. *Comput. Oper. Res.* **1997**, *24*, 1097–1100. [CrossRef]
43. Jatschka, T.; Rodemann, T.; Raidl, G.R. VNS and PBIG as Optimization Cores in a Cooperative Optimization Approach for Distributing Service Points. In *Computer Aided Systems Theory—EUROCAST 2019*; Moreno-Díaz, R., Pichler, F., Quesada-Arencibia, A., Eds.; Springer: Berlin/Heidelberg, Germany, 2020; Volume 12013, pp. 255–262.
44. Bouamama, S.; Blum, C.; Boukerram, A. A Population-based Iterated Greedy Algorithm for the Minimum Weight Vertex Cover Problem. *Appl. Soft Comput.* **2012**, *12*, 1632–1639. [CrossRef]
45. Church, R.; ReVelle, C. The maximal covering location problem. In *Papers in Regional Science*; Springer: Berlin/Heidelberg, Germany, 1974; Volume 32, pp. 101–118.
46. Farahani, R.Z.; Asgari, N.; Heidari, N.; Hosseininia, M.; Goh, M. Covering problems in facility location: A review. *Comput. Ind. Eng.* **2012**, *62*, 368–407. [CrossRef]
47. Richtárik, P.; Takáč, M. Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function. *Math. Program.* **2014**, *144*, 1–38. [CrossRef]