A Large Neighborhood Search for Battery Swapping Station Location Planning for Electric Scooters

Thomas Jatschka, Matthias Rauscher, Tobias Rodemann, Guenther Raidl

2023

Preprint:

This is a post-peer-review, pre-copyedit version of an article published in EuroCAST Conference 2022. The final authenticated version is available online at: https://doi.org/10.1007/978-3-031-25312-6_14

A Large Neighborhood Search for Battery Swapping Station Location Planning for Electric Scooters*

Thomas Jatschka[1], Matthias Rauscher[1], Bernhard Kreutzer[1], Yusuke Okamoto[2], Hiroaki Kataoka[2], Tobias Rodemann[3], and Günther R. Raidl[1]

[1] Institute of Logic and Computation, TU Wien, Austria
{tjatschk,raidl}@ac.tuwien.ac.at, e1527543@student.tuwien.ac.at, e0927086@student.tuwien.ac.at
[2] Honda R&D, Japan
{yusuke_01_okamoto,hiroaki_kataoka}@jp.honda
[3] HRI Europe, Germany
tobias.rodemann@honda-ri.de

Abstract. We consider the Multi Objective Battery Swapping Station Location Problem (MOBSSLP) for planning the setup of new stations for exchanging depleted batteries of electric scooters with the aim of minimizing a three-part objective function while satisfying an expected amount of demand. Batteries returned at a station are charged and provided to customers again once they are full. We present a large neighborhood search (LNS) for solving MOBSSLP instances. The LNS makes use of a mixed integer linear program (MILP) to quickly find good solutions within a specified neighborhood. Multiple neighborhood structures given by pairs of destroy and repair operators are suggested. The proposed LNS is evaluated on instances generated by adapted approaches from the literature with up to 500 potential station locations and up to 1000 user trips. Solutions obtained from the LNS have on average ten to thirty percent better objective values on these instances than a state-of-the-art MILP solver.

Keywords: Facility location problem · battery swapping stations · mixed integer linear programming · large neighborhood search

1 Introduction

A major hindrance for the large-scale adoption of electric vehicles (EVs) are the long battery recharging times. Especially for electric scooters, an attractive alternative to recharging depleted batteries is to replace them at dedicated stations. Once a depleted battery is returned to the station, the battery gets recharged and can then be made accessible to other customers again when fully charged.

* This project was partially funded by Honda Research Institute Europe and Honda R&D Co., Ltd.

In this work, we introduce the Multi Objective Battery Swapping Station Location Problem (MOBSSLP) and propose a large neighborhood search (LNS) [5] for solving it. In the MOBSSLP the task is to plan the setup of new stations for exchanging batteries of electric scooters or to extend existing stations with the aim of minimizing three different objectives combined in a linear fashion while satisfying an expected demand. The number of batteries a station can contain is decided by the number of battery modules assigned to the station. Battery swapping stations can be set up at dedicated locations which may differ in the maximum number of modules that can be added, opening times at which customers can exchange batteries, as well as setup and charging costs.

The MOBSSLP can be classified as a location-allocation problem [1] and is closely related to the capacitated multiple allocation fixed-charge facility location problem [2]. Moreover, the MOBSSLP is an adaption of the Multi-Period Battery Swapping Station Location Problem [3] in which customers are considered in an aggregated fashion, allowing better scalability to large numbers of customers and potential locations for stations.

For each of the three objectives of the MOBSSLP destroy and repair operators are presented. Additionally, we show how these operators can be effectively combined to consider all parts of the objective function together in the LNS. The LNS is implemented as a matheuristic [4] in which the repair operators make use of a mixed integer linear program (MILP).

We experimentally evaluate the proposed LNS on instances generated by adapted approaches from the literature. Results show that the LNS can outperform a general-purpose MILP solver, achieving solutions with objective values that are up to thirty percent smaller for instances with up to 500 potential station locations and up to 1000 user trips.

This work is based on parts on a master thesis [6], where more details and further results can be found.

2 Multi Objective Battery Swapping Station Location Problem

In the *Multi Objective Battery Swapping Station Location Problem* (MOBSSLP) the task is to plan the setup of stations for exchanging batteries of electric scooters or to extend already existing stations. We aim to minimize three different objectives combined in a linear fashion while satisfying a given demand in expectation. The three objectives are the setup cost for additional stations and extension modules, the cost for charging batteries, and the total duration of detours for users to exchange batteries. We consider a time horizon that is discretized into equally long consecutive time intervals represented by $\mathcal{T} = \{1, \dots, t_{\max}\}$. Moreover, we assume the planning horizon to be cyclic, i.e., the predecessor of the first interval is the last one and the successor of the last one the first interval.

Battery swapping stations can be set up at any of n different locations $L = \{1, \dots, n\}$. The costs for setting up a station at a location $l \in L$ with $s_l^{\text{ini}} \in \mathbb{N}$ initial battery slots are given by $c_l \geq 0$. One can add up to $e_l^{\max} \in \mathbb{N}$ additional battery exchange (BEX) modules with capacity $s^{\text{modul}} \in \mathbb{N}$ to the station at

location l for a cost of $c_l^{\text{modul}} \geq 0$ per module. Due to production limitations, the number of total BEX modules available is restricted, i.e., $z^{\text{modules}} \in \mathbb{N}$ refers to the maximum number of available BEX modules. Customers can exchange batteries at l in the time intervals $\mathcal{T}_l^{\text{ex}} \subseteq \mathcal{T}$. A battery that is returned to a station is recharged in the subsequent t^{c} time intervals. We distinguish between daytime and nighttime charging costs $c_l^{\text{dch}} \geq 0$ and $c_l^{\text{nch}} \geq 0$, respectively, per time interval with daytime referring to the set of time intervals $\mathcal{T}^{\text{dch}} \subseteq \mathcal{T}$.

Customer travel demands are given for origin-destination (O/D) pairs Q; let $m = |Q|$ be the number of these O/D pairs and $w_q^l \geq 0$ be the expected detour time for the O/D pair $q \in Q$ when making a fastest possible detour to location $l \in L$ for exchanging batteries there. Let $\mathcal{I} \subset \mathbb{N}$ be the set of vehicle types we consider represented by the corresponding numbers of batteries. We assume that batteries are all of the same type. The expected number of users with vehicle type $i \in \mathcal{I}$ who need to change batteries on trip $q \in Q$ during a time interval $t \in \mathcal{T}$ is denoted as d_{qi}^t.

A solution is primarily given by $x = (x_l)_{l \in L} \in \{0,1\}^n$ and $y = (y_l)_{l \in L}$ with $y_l \in \{0, \dots, e_l^{\max}\}$, where $x_l = 1$ indicates that a swapping station is to be set up at location l and y_l is the corresponding number of additionally installed BEX modules. Additionally, let assignment variables a_{qli}^t denote the part of the expected demand of O/D pair $q \in Q$ w.r.t. vehicle type $i \in \mathcal{I}$ which we assign to a location $l \in L$ during time interval $t \in \mathcal{T}_l^{\text{ex}}$.

We express the MOBSSLP by the following MILP.

$$\begin{aligned}
\min\ & \alpha_{\text{setup}} \sum_{l \in L} (c_l x_l + c_l^{\text{modul}} y_l) + \\
& \alpha_{\text{charging}} \sum_{l \in L} \sum_{q \in Q} \sum_{i \in \mathcal{I}} \sum_{t \in \mathcal{T}_l^{\text{ex}}} c_{lt}^{\text{ch}} \cdot i \cdot a_{qli}^t + \qquad (1) \\
& \alpha_{\text{delay}} \sum_{l \in L} \sum_{q \in Q} w_q^l \cdot \sum_{t \in \mathcal{T}_l^{\text{ex}}} \sum_{i \in \mathcal{I}} a_{qli}^t \\
& e_l^{\max} \cdot x_l \geq y_l \qquad \forall l \in L \quad (2) \\
& \sum_{l \in L | t \in \mathcal{T}_l^{\text{ex}}} a_{qli}^t \leq d_{qi}^t \qquad \forall t \in \mathcal{T},\ i \in \mathcal{I},\ q \in Q \quad (3) \\
& \sum_{t' \in \mathcal{T}_l^{\text{ch}}(t) \cup \{t\}} \sum_{q \in Q} \sum_{i \in \mathcal{I}} i \cdot a_{qli}^{t'} \leq s_l^{\text{ini}} x_l + s^{\text{modul}} y_l \qquad \forall l \in L,\ t \in \mathcal{T}_l^{\text{ex}} \quad (4) \\
& \sum_{q \in Q} \sum_{l \in L} \sum_{t \in \mathcal{T}_l^{\text{ex}}} \sum_{i \in \mathcal{I}} i \cdot a_{qli}^t = \sum_{q \in Q} \sum_{t \in \mathcal{T}_l^{\text{ex}}} \sum_{i \in \mathcal{I}} d_{qi}^t \qquad (5) \\
& \sum_{l \in L | c_l > 0} x_l + \sum_{l \in L} y_l \leq z^{\text{modules}} \qquad (6) \\
& x_l \in \{0,1\} \qquad \forall l \in L \quad (7) \\
& y_l \in \{0, \dots, e_l^{\max}\} \qquad \forall l \in L \quad (8) \\
& 0 \leq a_{qli}^t \leq \min\left(\frac{s_l^{\text{ini}} + e_l^{\max} \cdot s^{\text{modul}}}{i}, d_{qi}^t \right) \qquad \forall l \in L,\ t \in \mathcal{T}_l^{\text{ex}},\ i \in \mathcal{I},\ q \in Q \quad (9)
\end{aligned}$$

The objective function (1) is the linear combination of the different objectives with weights $\alpha_{\text{setup}} > 0$, $\alpha_{\text{charging}} > 0$ and $\alpha_{\text{delay}} > 0$. Inequalities (2) link variables x_l and y_l. Constraints (3) limit the amount of demand that can be assigned to the stations at each time interval. Inequalities (4) calculate the required capacity of a station at each location with $\mathcal{T}_l^{\text{ch}}(t)$ referring to the t^{c} subsequent time intervals succeeding $t \in \mathcal{T}$. Equality (5) ensures that all demand is satisfied. Constraint(6) restricts the total number of used BEX modules.

3 Large Neighborhood Search

In this section we present a large neighborhood search (LNS) based on the LNS presented in [3] for solving MBSSLP instances. Let (x, y, a) be a solution to the MOBSSLP. Moreover, let $L_0(x) \subseteq L$ be the set of locations with closed stations in x and $L_1(x) \subseteq L$ be the set of locations with open stations in x. In each iteration of the LNS, while the termination criterion has not yet been reached, a set of ν locations $L_{\text{destroy}} \subseteq L_1(x)$ is selected and destroyed by setting the number of modules to zero and un-allocating all associated demand. Afterwards, a repair procedure is applied to make the solution feasible again. For this purpose, first a set of ν' locations $L'_{\text{repair}} \subseteq L_0(x) \setminus L_{\text{destroy}}$ is selected. To generate the final repair set, we also add all locations in L_{destroy}, i.e. $L_{\text{repair}} = L'_{\text{repair}} \cup L_{\text{destroy}}$, to guarantee that we can always obtain a feasible solution. A solution is repaired w.r.t. a residual instance I of the original instance that only considers the demands of O/D pairs not assigned in the current partial solution. We first solve a relaxation of the MILP (1) to (9) in which we consider the y variables to be continuous. From the obtained solution which we denote with $(x, \tilde{y}, a)$ we then derive a feasible MOBSSLP solution as follows: First, all fractional $\tilde{y}$ values are rounded up, i.e., $y = (\lceil \tilde{y}_l \rceil)_{l \in L}$. Next, we greedily delete modules if the solution contains more than z_{modules} modules. Modules are deleted from locations $l \in L$ for which $\tilde{y}_l - \lfloor y_l \rfloor$ is the lowest. There may exist stations at locations $l \in L$ for which $s_l^{\text{ini}} < s^{\text{modul}}$. Removing such modules may result in an insufficient number of battery slots for satisfying the necessary demand. In such a case we iteratively close a random station and randomly add an equivalent number of modules to the remaining locations in the solution. This procedure is repeated until the total number of battery slots corresponds to the number of battery slots of the relaxed solution $(x, \tilde{y}, a)$. Finally, the demand is redistributed using the MILP (1) – (9) with the values of all x and y variables being fixed. For further details on how to repair a partial solution we refer to [6].

Next we present various selection operators for deciding which locations should be considered during the destroy and repair process. The operators are randomized greedy procedures that select locations according to their (potential) impact on the objective value w.r.t. to one or more objective goals. Moreover, locations are selected via tournament selection, i.e., to select one location a set of k (a strategy parameter) random candidate locations is first chosen randomly from $L_1(x)$ or $L_0(x)$, respectively. Then from this set the most promising candidate is chosen according to a criterion different for each selection operator and added to L_{destroy} or L'_{repair}, respectively. For the destroy selection schemes this

procedure is repeated ν times and ν' times for repair selection schemes, where ν and ν' are further strategy parameters.

For each objective we define one destroy and one repair selection operator. For the *Construction-Based Destroy Operator* the most promising location l that is added to L_{destroy} in each iteration is the candidate for which $\delta_l^{\text{setup}} = \frac{c_l + c_l^{\text{modul}} y_l}{s_l^{\text{ini}} + s_l^{\text{modul}} y_l}$ is the highest. For the *Delay-Based Destroy Operator* the most promising candidate l is the location with highest δ_l^{delay} as specified by Equation (10). Finally, for the *Delay-Based Charging Operator* the most promising candidate l is the location with highest δ_l^{ch} as specified by Equation (11).

$$\delta_l^{\text{delay}} = \frac{\sum_{q \in Q} \left(w_q^l \sum_{i \in \mathcal{I}} \sum_{t \in \mathcal{T}_l^{\text{ex}}} a_{qli}^t \right)}{\sum_{q \in Q} \sum_{i \in \mathcal{I}} \sum_{t \in \mathcal{T}_l^{\text{ex}}} a_{qli}^t} \quad (10) \qquad \delta_l^{\text{ch}} = \frac{\sum_{q \in Q} \sum_{i \in \mathcal{I}} \sum_{t \in \mathcal{T}_l^{\text{ex}}} c_{lt}^{\text{ch}} \cdot i \cdot a_{qli}^t}{\sum_{q \in Q} \sum_{i \in \mathcal{I}} \sum_{t \in \mathcal{T}_l^{\text{ex}}} i \cdot a_{qli}^t} \quad (11)$$

The repair selection operators select promising candidates in a similar way. More specifically, the *Construction-Based Repair Operator* chooses the location l for which ρ_l^{setup}, given by Equation (12), is the lowest. The *Delay-Based Repair Operator* selects the candidate for which ρ_l^{delay}, given by Equation (14), is the lowest. Finally, the *Charging-Based Repair Operator* selects the candidate for which ρ_l^{ch}, given by Equation (15), is the lowest.

$$\rho_l^{\text{setup}} = \frac{c_l + c_l^{\text{modul}} \min(y_{\text{avg}}, e_l^{\max})}{s_l^{\text{ini}} + s_l^{\text{modul}} \min(y_{\text{avg}}, e_l^{\max})} \quad (12) \qquad y_{\text{avg}} = \frac{\sum_{l \in L_{\text{destroy}}} y_l}{\nu} \quad (13)$$

$$\rho_l^{\text{delay}} = \frac{\sum_{q \in Q} \left(w_q^l \sum_{i \in \mathcal{I}} \sum_{t \in \mathcal{T}_l^{\text{ex}}} d'^t_{qi} \right)}{\sum_{q \in Q} \sum_{i \in \mathcal{I}} \sum_{t \in \mathcal{T}_l^{\text{ex}}} d'^t_{qi}} \quad (14) \qquad \rho_l^{\text{ch}} = \frac{\sum_{q \in Q} \sum_{i \in \mathcal{I}} \sum_{t \in \mathcal{T}_l^{\text{ex}}} c_{lt}^{\text{ch}} \cdot i \cdot d'^t_{qi}}{\sum_{q \in Q} \sum_{i \in \mathcal{I}} \sum_{t \in \mathcal{T}_l^{\text{ex}}} i \cdot d'^t_{qi}} \quad (15)$$

For the *Delay-Based Repair Operator* and the *Charging-Based Repair Operator* we also estimate which demands are potentially covered by a selected candidate l and do not consider these demands in the remaining steps of the selection procedure anymore. For further details we refer to [6].

The presented selection operators can also be combined to select locations according to multiple objective goals. In the most straight forward way, in each iteration of the LNS the repair and destroy operators are selected randomly, choosing from the above presented selection operators. We refer to these operators as *Mixed Destroy Operator* and *Mixed Repair Operator*, respectively.

In a more sophisticated way, the *Weighted Sum Destroy Operator* chooses the candidate l with the highest impact on the objective function, i.e., the largest value $\alpha_{\text{setup}} \cdot \delta_l^{\text{setup}} + \alpha_{\text{delay}} \cdot \delta_l^{\text{delay}} + \alpha_{\text{charging}} \cdot \delta_l^{\text{ch}}$.

Similarly, the *Weighted Sum Repair Operator* selects the candidate location l for which the estimated impact on the objective function is the lowest represented by the value $\alpha_{\text{setup}} \cdot \rho_l^{\text{setup}} + \alpha_{\text{delay}} \cdot \rho_l^{\text{delay}} + \alpha_{\text{charging}} \cdot \rho_l^{\text{ch}}$. Finally, the *Objective-Based Repair Operator* uses the same procedure as the delay- and charging-

based repair operator to prevent already covered demand from being considered in future iterations of the selection procedure.

4 Computational Results

We test our LNS on artificial instances with properties chosen based on information provided by Honda R&D. We created six groups of instances identified by their number of station locations n and number of O/D pairs m as (n, m). For each subgroup we generate 30 instances. For more details on how the instances were generated, see [6]. On each instance, we test three different alpha configurations which differ in the α_{delay} parameter, i.e. $\alpha_{\text{charging}} = 0.01$, $\alpha_{\text{setup}} = 0.01$, and $\alpha_{\text{delay}} \in \{0.1, 1, 10\}$. Therefore, in the remainder of this section each configuration will be identified only by α_{delay}. For the parameters of the LNS we determined $\nu = \nu' = 5$ and $k = 5$ in preliminary experiments. For the procedure for constructing an initial solution for the LNS we refer to [6].

Table 1: Average optimality gaps, for different α_{delay} for each selection strategy.

	gap (%)								
	$\alpha_{\text{delay}} = 0.1$			$\alpha_{\text{delay}} = 1.0$			$\alpha_{\text{delay}} = 10.0$		
(n, m)	*constr*	*delay*	*charging*	*constr*	*delay*	*charging*	*constr*	*delay*	*charging*
(50, 100)	2.73	3.01	**2.61**	6.54	**5.60**	5.76	12.77	**10.94**	12.05
(100, 200)	2.77	**1.97**	2.80	6.69	**5.61**	6.81	22.41	**18.27**	25.82
(200, 400)	**4.49**	5.72	5.49	**17.43**	18.65	21.31	41.89	**36.78**	47.47
(300, 600)	**5.13**	6.88	6.31	**28.41**	29.13	32.32	62.37	**59.42**	67.99
(400, 800)	**6.50**	8.62	8.39	**33.75**	33.96	36.63	71.48	**70.21**	74.49
(500, 1000)	**7.98**	10.77	10.68	**36.16**	37.03	39.99	74.80	**74.25**	77.59

	gap (%)					
	$\alpha_{\text{delay}} = 0.1$		$\alpha_{\text{delay}} = 1.0$		$\alpha_{\text{delay}} = 10.0$	
(n, m)	*mixed*	*wsum*	*mixed*	*wsum*	*mixed*	*wsum*
(50, 100)	2.51	**2.42**	**5.84**	6.50	**8.87**	11.87
(100, 200)	2.72	**2.60**	**5.84**	6.06	**17.71**	20.15
(200, 400)	**3.34**	4.75	17.49	**17.30**	**38.52**	41.39
(300, 600)	5.07	**4.84**	**27.35**	28.35	**60.98**	62.21
(400, 800)	**6.59**	6.84	32.79	**32.78**	**70.15**	70.33
(500, 1000)	**8.16**	8.22	**36.30**	36.60	74.65	**74.06**

The proposed algorithms were implemented in Julia[4] 1.6.1 using the JuMP package[5] and Gurobi[6] 9.1 as underlying MILP solver. All test runs have been executed on an Intel Xeon E5-2640 v4 2.40GHz machine in single-threaded mode with a global time limit of one hour per run.

We investigate five different strategies for selecting the locations considered in the destroy and repair procedures: *constr*, *delay* and *charging* use only the

[4] https://julialang.org/
[5] https://jump.dev/JuMP.jl/stable/
[6] https://www.gurobi.com/

construction-, delay- and charging-based destroy and repair operators, respectively. Moreover, *mixed* and *wsum* use only the mixed selection operators and weighted sum selection operators, respectively. We evaluate the quality of solutions in terms of optimality gaps to the best lower bounds obtained by trying to solve the MILP given by Equations (1)–(9) within the one hour time limit. Table 1 shows average optimality gaps, for all considered α_{delay} configurations. We can see that optimality gaps generally increase with growing instance size and growing α_{delay} value for all five strategies. As expected, we can observe that *delay* performs better, the higher α_{delay} is, i.e., as minimizing the delay becomes more important LNS operators destroying and repairing stations based on their induced delay produce better results. For lower values of α_{delay} *constr* shows the best performance. Regarding the multi-objective strategies, one can see that for $\alpha_{\text{delay}} = 0.1$ and $\alpha_{\text{delay}} = 1.0$ *mixed* is slightly more favorable. For $\alpha_{\text{delay}} = 10.0$ the difference becomes more evident, as *mixed* achieves up to 3% better results. In general the performance of the multi-objective strategies is comparable to the performance of the best single-objective strategy for each α_{delay}. Therefore, as expected, multi-objective strategies are more robust to changes of the weights of the objective function.

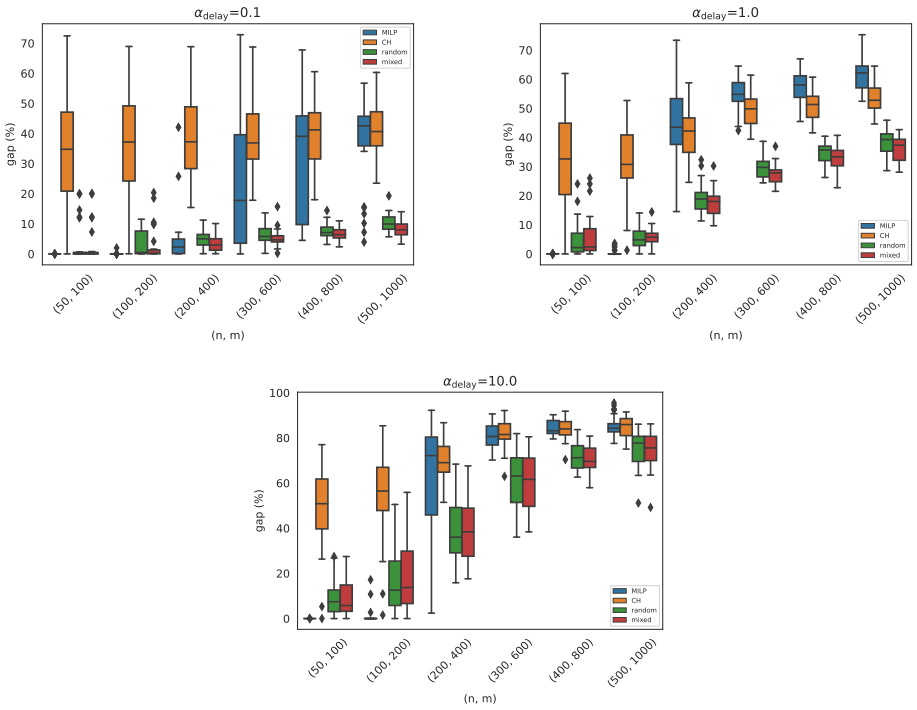

Fig. 1: Final optimality gaps when solving the MOBSSLP with different approaches.

Finally, Figure 1 compares the LNS with selection strategy *mixed* to other approaches. In particular, we compare the obtained results to those of an LNS using a uniform random selection strategy, denoted by *random*, to solutions generated by the initial construction heuristic, denoted by CH, as well as solutions obtained by Gurobi within one hour w.r.t. the MILP (1) – (9), denoted by MILP. Starting from (200, 400) our LNS approach is able to consistently achieve superior results with up to 29% lower objective values than those obtained by the MILP approach. Moreover, we can see that the LNS strongly improves the initial solution obtained by the construction heuristic. Finally, we performed one-sided Wilcoxon signed-rank tests between the solutions obtained by *mixed* and the solutions obtained by *random*. For almost all instance groups and values of α_{delay}, *mixed* achieved significantly better results than *random* within a 95% confidence interval.

5 Conclusion and Future Work

We considered the *Multi Objective Battery Swapping Station Location Problem* (MOBSSLP) and presented a large neighborhood search (LNS) using mixed integer linear programming (MILP) for repairing solutions. We proposed different strategies for selecting the locations for the destroy and repair procedure of the LNS. For larger instances our evaluation shows that our LNS far surpasses the performance of a state-of-the-art MILP solver in terms of solution quality. We have further seen that combining all single objective selection strategies results in a strategy more robust to changes in the weighting of the objective function.

For future work it seems promising to extend the approach to an Adaptive Large Neighborhood Search (ALNS) which then dynamically selects the most promising selection operator in each iteration. Moreover, it could also be interesting to research adapted variants of the MOBSSLP, considering aspects such as limited charging times or an overall budget for building stations and modules.

References

1. Boloori Arabani, A., Farahani, R.Z.: Facility location dynamics: An overview of classifications and applications. Computers & Industrial Engineering **62**(1), 408–420 (2012)
2. Farahani, R.Z., Hekmatfar, M.: Facility location: concepts, models, algorithms and case studies. Contributions to Management Science, Springer (2009)
3. Jatschka, T., Oberweger, F.F., Rodemann, T., Raidl, G.R.: Distributing battery swapping stations for electric scooters in an urban area. In: Olenev, N., Evtushenko, Y., Khachay, M., Malkova, V. (eds.) Optimization and Applications, Proceedings of OPTIMA 2020 – XI International Conference Optimization and Applications. LNCS, vol. 12422, pp. 150–165. Springer (2020)
4. Maniezzo, V., Stützle, T., Voß, S.: Matheuristics. Annals of Information Systems, Springer Nature, 1 edn. (2010)
5. Pisinger, D., Ropke, S.: Large neighborhood search. In: Gendreau, M., Potvin, J.Y. (eds.) Handbook of Metaheuristics, pp. 399–419. Springer (2010)
6. Rauscher, M.: A Matheuristic for Battery Exchange Station Location Planning for Electric Scooters. Master's thesis, TU Wien, Vienna, Austria (2022), https://catalogplus.tuwien.at/permalink/f/qknpf/UTW_alma71122610460003336