This is an accepted version of the article: Woller, D., Koz{\' a}k, V., and Kulich, M. (2020). **The GRASP Metaheuristic for the Electric Vehicle Routing Problem**. *Modelling and Simulation for Autonomous Systems*, which has been published in a final form at https://doi.org/10.1007/978-3-030-70740-8_12.

The GRASP Metaheuristic for the Electric Vehicle Routing Problem

David Woller(✉), Viktor Kozák and Miroslav Kulich

Czech Institute of Informatics, Robotics, and Cybernetics,
Czech Technical University in Prague, Jugoslávských partyzánů 1580/3 160 00 Praha 6, Czech Republic
{wolledav,viktor.kozak,kulich}@cvut.cz
http://imr.ciirc.cvut.cz

Abstract. The Electric Vehicle Routing Problem (EVRP) is a recently formulated combination of the Capacitated Vehicle Routing Problem (CVRP) and the Green Vehicle Routing Problem (GVRP). The goal is to satisfy all customers' demands while considering the vehicles' load capacity and limited driving range. All vehicles start from one central depot and can recharge during operation at multiple charging stations. The EVRP reflects the recent introduction of electric vehicles into fleets of delivery companies and represents a general formulation of numerous more specific VRP variants. This paper presents a newly proposed approach based on Greedy Randomized Adaptive Search Procedure (GRASP) scheme addressing the EVRP and documents its performance on a recently created dataset. GRASP is a neighborhood-oriented metaheuristic performing repeated randomized construction of a valid solution, which is subsequently further improved in a local search phase. The implemented metaheuristic improves multiple best-known solutions and sets a benchmark on some previously unsolved instances.

Keywords: electric vehicle routing problem, greedy randomized adaptive search, combinatorial optimization

1 Introduction

This paper addresses the Electric Vehicle Routing Problem (EVRP) recently formulated in [14]. The EVRP is a challenging $\mathcal{NP}$-hard combinatorial optimization problem. It can be viewed as a combination of two variants of the classical Vehicle Routing Problem (VRP) - the Capacitated Vehicle Routing Problem (CVRP) and the Green Vehicle Routing Problem (GVRP). In the VRP, the goal is to minimize the total distance traveled by a fleet of vehicles/agents, while visiting each customer exactly once. In the CVRP, the customers are assigned an integer-valued positive demand, and the vehicles have limited carrying capacity. Thus, an additional constraint of satisfying all customers' demands while respecting the limited vehicle capacity is added to the VRP. Concerning the GVRP, the

terminology is not completely settled, but the common idea among different formulations aims at minimizing the environmental impact, typically by taking into consideration the limited driving range of alternative fuel-powered vehicles and the possibility of refueling at rarely available Alternative Fuel Stations (AFSs). The EVRP has the same objective as the VRP while incorporating the additional constraints from CVRP and GVRP. It is sometimes alternatively named CGVRP, while the name is EVRP often used for other variants of the GVRP (e.g., with considering the non-linear time characteristic of the recharging process or the influence of carried load on the energy consumption).

A method based on the Greedy Randomized Adaptive Search Procedure (GRASP) metaheuristic, together with the preliminary results, is presented in the paper. A novel dataset was introduced in [13], and it is the first publicly available dataset of EVRP instances. A subset of this dataset is used in the competition [12]. This paper is the first to give results to the whole dataset and thus represents a valuable benchmark for future researchers. As the competition results were not known at the time of writing, the results are compared only to the organizers-provided best-known values for the smallest instances.

1.1 Related works

Various approaches were successfully applied to the numerous variants of the VRP, and many of these can also be adapted to the EVRP formulation solved. For example, a recent survey [5] focused only on the variants of EVRP presented a total number of 79 papers. Most of these consider additional constraints intended to reflect real-world conditions, such as using heterogeneous or mixed vehicle fleet, hybrid vehicles, allowing partial recharging or battery swapping, considering different charging technologies and the non-linearity of charging function, dynamic traffic conditions, customer time windows, and others. The problem formulation introduced in [14] stands out, as it leaves out all but the most fundamental constraints on battery and load capacity. Thus, addressing it might produce a universal method adjustable for more specific variants.

According to [5], the most commonly applied metaheuristics are Adaptive Large Neighborhood Search (ALNS), Genetic Algorithms (GA), Large Neighborhood Search (LNS), Tabu Search (TS), Iterative Local Search (ILS) and Variable Neighborhood Search (VNS). The GRASP metaheuristic deployed in this paper is, therefore, not a commonly used one. However, it follows similar principles as other neighborhood-oriented metaheuristics, such as TS, ILS, or VNS. Exact methods such as Dynamic Programming or various Branch-and-Bound/Branch-and-Cut techniques are frequently used as well, but these are generally not suitable for solving larger instances in a reasonable time.

According to [14], the EVRP variant solved was first formulated in [9]. So far, only a few papers are dealing with this problem, and for each one of them, the exact formulation slightly varies. The first one is [18], which additionally limits the maximum number of routes. It presents a solution method based on the Ant Colony System (ACS) algorithm. The second one is [15], which considers the maximum total delivery time. It presents a Simulated Annealing (SA) algorithm

operating with four different local search operators (swap, insert, insert AFS, and delete AFS). Then, [17] proposes a novel construction method and a memetic algorithm consisting of a local search and an evolutionary algorithm. Similarly to [15], the local search combines the Variable Neighborhood Search (VNS) with the Simulated Annealing acceptance criterion. The operators used in the local search are 2-opt, swap, insert, and inverse. Unlike the previous approaches, the proposed algorithm is memetic, thus maintains a whole set of solutions. The organizers of [12] themselves also presented a solution method together with the new dataset in [13]. Similarly to [18], they employ an Ant Colony Optimization metaheuristic. However, the problem formulation in [13] differs from [14] and the previously mentioned methods, as it evaluates the energy consumption as a function of the current load. Thus, their results are not directly comparable.

2 Methods

2.1 Problem formulation

The EVRP can be described as follows: given a fleet of EVs, the goal is to find a route for each EV, such that the following requirements are met. The EVs must start and end at the central depot and serve a set of customers. The objective is to minimize the total distance traveled, while each customer is visited exactly once, for every EV route the total demand of customers does not exceed the EV's maximal carrying capacity and the EV's battery charge level does not fall below zero at any time. All EVs begin and end at the depot, EVs always leave the AFS fully charged (or fully charged an loaded, in case of the depot), and the AFSs (including the depot) can be visited multiple times by any EV. An example of a solved EVRP instance is shown in Figure 1. Here, the depot, the AFSs, and the customers are represented by a red circle, black squares, and blue circles, respectively. The blue line represents the planned EVRP tour.

The EVRP mathematical formulation as introduced in [14] follows.

$$\min \sum_{i \in V, j \in V, i \neq j} w_{ij} x_{ij}, \tag{1}$$

s.t.

$$\sum_{j \in V, i \neq j} x_{ij} = 1, \forall i \in I, \tag{2}$$

$$\sum_{j \in V, i \neq j} x_{ij} \leq 1, \forall i \in F', \tag{3}$$

$$\sum_{j \in V, i \neq j} x_{ij} - \sum_{j \in V, i \neq j} x_{ji} = 0, \forall i \in V, \tag{4}$$

$$u_j \leq u_i - b_i x_{ij} + C(1 - x_{ij}), \forall i \in V, \forall j \in V, i \neq j, \tag{5}$$

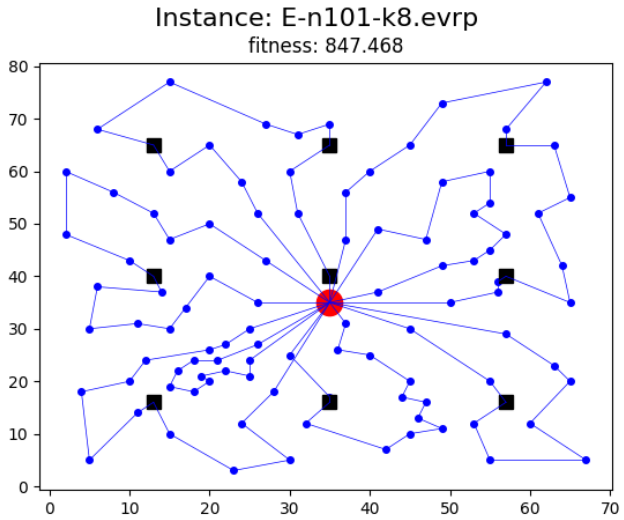


Fig. 1: Solved EVRP instance

$$0 \leq u_i \leq C, \forall i \in V, \tag{6}$$

$$y_j \leq y_i - hw_{ij}x_{ij} + Q(1 - x_{ij}), \forall i \in I, \forall j \in V, i \neq j, \tag{7}$$

$$y_j \leq Q - hw_{ij}x_{ij}, \forall i \in F' \cup D, \forall j \in V, i \neq j, \tag{8}$$

$$0 \leq y_i \leq Q, \forall i \in V, \tag{9}$$

$$x_{ij} \in \{0, 1\}, \forall j \in V, i \neq j, \tag{10}$$

where $V = \{D \cup I \cup F'\}$ is a set of nodes. Set I denotes the set of customers, set F' denotes set of β_i node copies of each AFS $i \in F$ (i.e., $|F'| = \sum_{i \in F} \beta_i$) and D denotes the central depot. Lets also define $E = \{(i, j)|i, j \in V, i \neq j\}$ as a set of edges in the fully connected weighted graph $G = (V, E)$. Then, x_{ij} is a binary decision variable corresponding to usage of the edge from node $i \in V$ to node $j \in V$ and w_{ij} is the weight of this edge. Variables u_i and y_i denote, respectively, the remaining carrying capacity and remaining battery charge flevel of an EV on its arrival at node $i \in V$. Finally, the constant h denotes the consumption rate of the EVs, C denotes their maximal carrying capacity, Q the maximal battery charge level, and b_i the demand of each customer $i \in I$.

For the purposes of formal components description, let's also define an EVRP tour T as a sequence of nodes $T = \{v_0, v_1, ..., v_{n-1}\}$, where v_i is a customer, a depot or an AFS and n is the length of the tour T. Finally, let

$$w(T) = \sum_{i=0}^{n-2} w_{i,i+1} \tag{11}$$

be the weight of the whole tour T.

2.2 GRASP Metaheuristic

GRASP is a well-established metaheuristic first introduced in [6] in 1989. Since then, it was successfully applied to numerous operations research problems, such as routing, covering and partition, location, minimum Steiner tree, optimization in graphs, assignment, and scheduling [7]. Its industrial applications include fields such as manufacturing, transportation, telecommunications, graph and map drawing, power systems, computational biology, or VLSI.

GRASP is a multi-start metaheuristic suitable for computing near-optimal solutions of combinatorial optimization problems. It is described in Algorithm 1. At the beginning, the best found tour T^* is initialized as empty and its weight $w(T^*)$ is set to infinity (lines 1-2). Then, the following process is repeated until a stop condition is met. A tour T visiting all customers is built from scratch, using a greedy randomized construction (line 4). If T is not a valid EVRP tour (e.g. constraints on load or battery capacity are not satisfied), tour T is fixed by a repair procedure (line 5-6). After that, the tour T is improved by a local search procedure (line 7), where a local minimum is found. The best tour found overall T^* is then updated, if T yields better weight (line 8-9). In this application, the stop condition is defined by a maximal number of GRASP iterations $MaxIters$.

Algorithm 1: Greedy Randomized Adaptive Search (GRASP)

Input: max. number of iterations $MaxIters$
Output: best tour found T^*

```
1  T* ← ∅
2  w(T*) ← ∞
3  for i = 1 to MaxIters do
4  |  T ← greedyRandomizedConstruction()
5  |  if !isValid(T) then
6  |  |  T ← repair(T)
7  |  T ← localSearch(T)
8  |  if w(T) < w(T*) then
9  |  |  T* ← T
10 return T*
```

2.3 Construction

In each iteration of the GRASP metaheuristic, a new valid EVRP tour is to be constructed. This tour serves as a starting point to the subsequent local search. According to the GRASP philosophy, a commonly used Nearest Neighbor (NN) heuristic was utilized for the greedy randomized construction. Due to the additional constraints imposed by the EVRP formulation, the NN construction can produce an invalid EVRP tour. Therefore, a repair procedure is needed. A novel procedure called Separate Sequential Fixing (SSF) was designed for this purpose.

Nearest Neighbor (NN) construction [10] is a commonly used algorithm to find an approximate solution to the Travelling Salesman Problem (TSP). The EVRP is equivalent to the TSP if the battery and load constraints are omitted. As these constraints cannot be easily incorporated into an iterative construction, it is convenient to determine only the order of the customers with the NN construction. The construction is described in Algorithm 2. The input to the algorithm is a set of all customers I, the depot D, and a set of edges E_{ID} in the fully connected weighted graph $G_{ID} = (D \cup I, E_{ID})$. Note that the AFSs F' and the corresponding edges are not considered in this phase. The output is then a TSP tour, which starts and ends at D and visits all customers. At the very beginning, the depot is added to T (line 1). Then, a first customer to be visited is randomly selected (line 3). After that, the remaining customers are greedily added to the tour one by one. Each time, the customer which is closest to the previously added customer is selected (line 5-8). Finally, the depot is added again and the tour is closed (line 9).

Algorithm 2: NN construction

Input: graph $G_{ID} = (D \cup I, E_{ID})$
Output: tour visiting all customers T_{TSP}

1 T_{TSP}.`append`(D)
2 Mark all customers in I as unvisited
3 Randomly select $c \in I$
4 T_{TSP}.`append`(c), mark c as visited
5 **while** all customers not visited **do**
6 | Find the shortest edge from c to an unvisited $c' \in I$
7 | T_{TSP}.`append`(c'), mark c' as visited
8 | $c \leftarrow c'$
9 T_{TSP}.`append`(D)
10 **return** T

Separate Sequential Fixing (SSF) repair procedure is a newly proposed method designed to repair such an EVRP tour, where the constraints on battery

or load capacity are not met. It consists of two phases, in which the constraints violations are fixed separately. If the following two assumptions are satisfied, the SSF procedure guarantees to produce a valid EVRP tour. First, the graph of all AFS and the depot must be connected. Second, each customer must be reachable from at least one AFS or depot.

The first phase of SSF is described in Algorithm 3. It takes a TSP tour over all of the customers T_{TSP} as an input and outputs a valid CVRP tour T_{CVRP}. All nodes are sequentially copied from the T_{TSP} to the T_{CVRP}, and the current vehicle load $CurLoad$ is held. If the current load is not sufficient for satisfying the next customer, the depot is added to the T_{CVRP} first.

Algorithm 3: SSF repair procedure - phase 1

Input: tour visiting all customers T_{TSP}
Output: valid CVRP tour T_{CVRP}

```
1  T_CVRP ← ∅
2  T_CVRP.append(T_TSP.popFront())
3  CurLoad ← MaxLoad
4  for Next in T_TSP do
5      if CurLoad ≥ demand(Next) then
6          T_CVRP.append(Next)
7          CurLoad ← CurLoad − demand(Next)
8      else
9          T_CVRP.append(D)
10         T_CVRP.append(Next)
11         CurLoad ← MaxLoad − demand(Next)
12 return T_CVRP
```

The second phase is described in Algorithm 4. It takes a T_{CVRP} as an input and outputs a valid EVRP tour T_{EVRP}. This time, the nodes are sequentially copied from the T_{CVRP} to the T_{EVRP}. Initially, the depot is added to the T_{EVRP} and the current battery level $CurBattery$ is set to maximum (line 2-3). Then, the following loop is performed for all of the remaining nodes in the T_{CVRP}. The last node already added to the T_{EVRP} is denoted as $Current$ (line 5). The next node to be added is denoted as $Next$, $NextBattery$ is the potential battery level in the $Next$ node (line 6-9), and $NextAFS$ is the AFS, which is closest to the $Next$ node (line 10). If the $Next$ node is directly reachable from $Current$ and the vehicle will not get stuck in it, $Next$ is added to T_{EVRP} (line 11-15). Otherwise, the $CurrentAFS$ node, which is the closest AFS to $Current$, is determined (line 17). Then, a sequence of AFSs from $CurrentAFS$ to $NextAFS$ is added to T_{EVRP} (line 18). This sequence is obtained as the shortest path on a graph of all AFSs and a depot. Due to the condition about not getting stuck (line 13) and the two SSF assumptions, $CurrentAFS$ is always reachable from $Current$. After $NextAFS$, $Next$ can be added as well, and the loop is repeated until the T_{CVRP} is not empty.

Algorithm 4: SSF repair procedure - phase 2

Input: valid CVRP tour T_{CVRP}
Output: valid EVRP tour T_{EVRP}

```
1  T_EVRP ← ∅
2  T_EVRP.append(T_CVRP.popFront())
3  CurBattery ← MaxBattery
4  for Next in T_CVRP do
5  |  Current ← T_EVRP.back()
6  |  if isAFS(Next) then
7  |  |  NextBattery ← MaxBattery
8  |  else
9  |  |  NextBattery ← CurBattery − getConsumption(Current, Next)
10 |  NextAFS ← getClosestAFS(Next)
11 |  Reachable ← CurBattery ≥ getConsumption(Current, Next)
12 |  Stuck ← NextBattery < getConsumption(Next, NextAFS)
13 |  if Reachable & !Stuck then
14 |  |  T_EVRP.append(Next)
15 |  |  CurBattery ← NextBattery
16 |  else
17 |  |  CurAFS ← getClosestAFS(Current)
18 |  |  T_EVRP.append(getPath(CurAFS, NextAFS))
19 |  |  T_EVRP.append(Next)
20 |  |  CurBattery ← MaxBattery − getConsumption(T_EVRP.back(), Next)
21 return T_EVRP
```

2.4 Local Search

This section provides a detailed description of the local search that is performed within the GRASP metaheuristic described in Section 2.2. The local search uses several local search operators, corresponding to different neighborhoods of an EVRP tour. The application of these operators is controlled by a simple heuristic. For this purpose, the Variable Neighborhood Descent (VND) and its randomized variant (RVND) were selected [3].

(Randomized) Variable Neighborhood Descent - (R)VND is a heuristic commonly used as a local search routine in other metaheuristics. It has a deterministic variant (VND) and a stochastic one (RVND). Both variants are described in Algorithm 5.

The input is a valid EVRP tour T and a sequence of local search operators $\mathcal{N}$, corresponding to different neighborhoods in the search space. The output is a potentially improved valid EVRP tour T. Both of the heuristic variants perform the local search sequentially in the neighborhoods in $\mathcal{N}$. In the case of the RVND, the sequence of the neighborhoods is randomly shuffled first (line 2), whereas, in the VND, the order remains fixed. Then, the heuristic attempts to improve the current tour T in the i-th neighborhood $\mathcal{N}_i$ according to the best

improvement scenario (line 4). This corresponds to searching the local optimum in $\mathcal{N}_i(T)$. Each time an improvement is made, the local search is restarted and T is updated accordingly (line 5-7). The VND then starts again from the first neighborhood in $\mathcal{N}$, while the RVND randomly reshuffles the neighborhoods first (line 8). The algorithm terminates when no improvement is achieved in any of the neighborhoods.

Algorithm 5: (Rand.) Variable Neighborhood Descent - (R)VND

Input: valid EVRP tour T, neighborhoods $\mathcal{N}$
Output: potentially improved valid EVRP tour T

```
1  i ← 1
2  Randomly shuffle 𝒩 // RVND only
3  while i ≤ |𝒩| do
4  |  T' ← arg min_{T̃ ∈ 𝒩_i(T)} w(T̃)
5  |  if w(T') < w(T) then
6  |  |  T ← T'
7  |  |  i ← 1
8  |  |  Randomly shuffle 𝒩 // RVND only
9  |  else
10 |  |  i ← i + 1
11 return T
```

Local Search Operators. A description of individual local search operators corresponding to different neighborhoods follows. Several operators commonly used in problems, where the solution can be encoded as a permutation (e.g., the TSP), were adapted for the EVRP. These operators are the 2-opt [4] and 2-string, which is a generalized version of numerous other commonly used operators.

An essential part of a neighborhood-oriented local search is efficient cost update computation. As both the 2-string derived operators and the 2-opt take two input parameters, the time complexity of exploring the whole neighborhood is $\mathcal{O}(n^2)$, where $n = |V|$. A naive approach is to apply every possible combination of parameters, create a modified tour $\tilde{T}$, and determine its weight $w(\tilde{T})$ in order to discover the most improving move. However, evaluating $w(\tilde{T})$ is a $\mathcal{O}(n)$ operation, thus the time complexity of the local search in each neighborhood would become $\mathcal{O}(n^3)$. This could be prevented by deriving $\mathcal{O}(1)$ cost update functions δ, which can be expressed as a difference between the sum of removed edges weights and the sum of newly added edges weights. Thus, a positive value of the cost update corresponds to an improvement in fitness and vice versa. As the operators can produce an invalid EVRP tour, each local optimum candidate $\tilde{T} \in \mathcal{N}_i(T)$ is determined and checked for validity before acceptance as T'.

2-opt is an operator commonly used in many variants of classical planning problems such as TSP or VRP. It takes a pair of indices i, j, and a tour T as an

input and returns a modified tour $\tilde{T}$, where the sequence of nodes from i-th to j-th index is reversed. It must hold, that $i < j$, $i \geq 0$ and $j < n$.

The cost update function δ_{2-opt} can be evaluated as

$$\delta_{2-opt} = w_{i-1,i} + w_{j,j+1} - w_{i-1,j} - w_{i,j+1}, \tag{12}$$

where the indices are expressed w.r.t. to the tour T.

2-string and its variants is a generalized version of several commonly used operators, which can be obtained by fixing some of the 2-string parameters. The 2-string operator takes five parameters: a tour T, a pair of indices i, j valid w.r.t. to T, and a pair of non-negative integers X, Y. It returns a modified tour $\tilde{T}$, where the sequence of X nodes following after the i-th node in T is swapped with the sequence of Y nodes following after the j-th node. It must hold, that $i \geq 0$, $j \geq i + X$ and $j + Y \leq n - 1$. The following operators can be derived by fixing the values of X and Y:

- 1-point: $X = 0, Y = 1$
- 2-point: $X = 1, Y = 1$
- 3-point: $X = 1, Y = 2$
- or-opt2: $X = 0, Y = 2$
- or-opt3: $X = 0, Y = 3$
- or-opt4: $X = 0, Y = 4$
- or-opt5: $X = 0, Y = 5$

When performing the local search, the complementary variants of these operators (e.g., 1-point with $X = 1, Y = 0$) are considered as well.

The cost update function $\delta_{2-string}$ can be evaluated as

$$\delta_{2-string} = cut_1 + cut_2 + cut_3 + cut_4 - add_1 - add_2 - add_3 - add_4, \tag{13}$$

where cut_1 corresponds to the edge weight after i-th node in T, cut_2 to the edge after $i + X$, cut_3 to the edge after j and cut_4 to the edge after $j + Y$. Then, add_1 is the weight of the edge added after the index i-th node in T, add_2 of the edge added after the reinserted block of Y nodes, add_3 of the edge added after j and add_4 of the edge added after the reinserted block of the X nodes. For some combinations of the parameters, some of these values evaluate to zero, which must be carefully treated in the implementation. For example, if $X \neq 0$, then $cut_2 = w_{i+X,i+X+1}$, otherwise $cut_2 = 0$.

3 Results and Discussion

This section documents the parameters tuning and evaluates the performance of the proposed GRASP metaheuristic on the dataset introduced in [13]. Note that the CGVRP formulation used in [13] slightly differs from the EVRP formulation used in this paper and [14]. In [13], the energy consumption depends on the

traveled distance and the current vehicle load, whereas in [14], it depends only on the traveled distance. Thus, the best-known values provided in [13] are not relevant when using the more general formulation from [14].

The stop condition of the metaheuristic is also adopted from [14]. An individual run terminates after $25000n$ fitness evaluations of a tour T, where $n = |V|$ is the actual problem size, that is, the total number of unique nodes. A single call to a distance matrix is counted as $1/n$ of an evaluation. Consistently with [14], each problem instance is solved 20 times, with random seeds ranging from 1 to 20. The experiments were carried out on a computer with an Intel Core i7-8700 3.20GHz processor and 32 GB RAM.

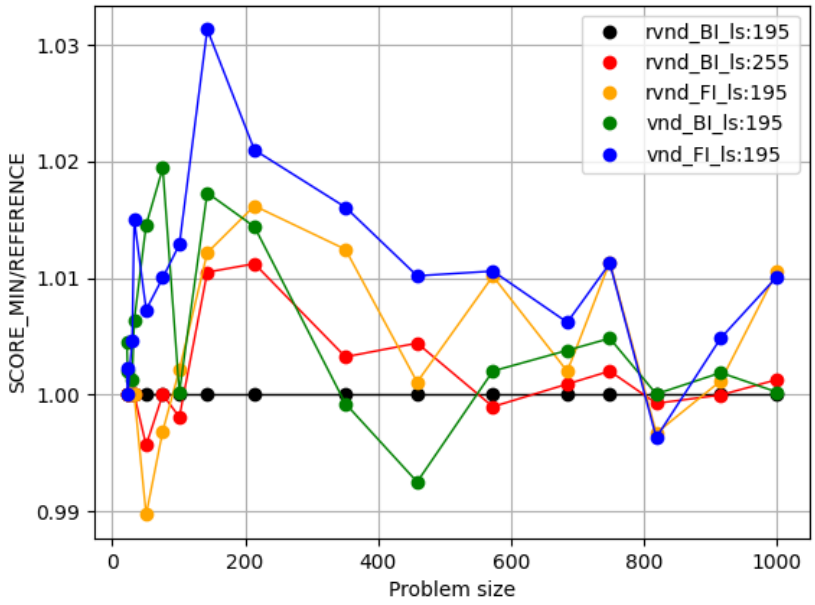
The rest of this section is structured as follows. Section 3.1 provides detailed information about the used dataset [13]. The process of tuning GRASP parameters on a subset of the dataset is described in Section 3.2. The final results of the tuned metaheuristic on the whole dataset are given in Section 3.3.

3.1 Dataset description

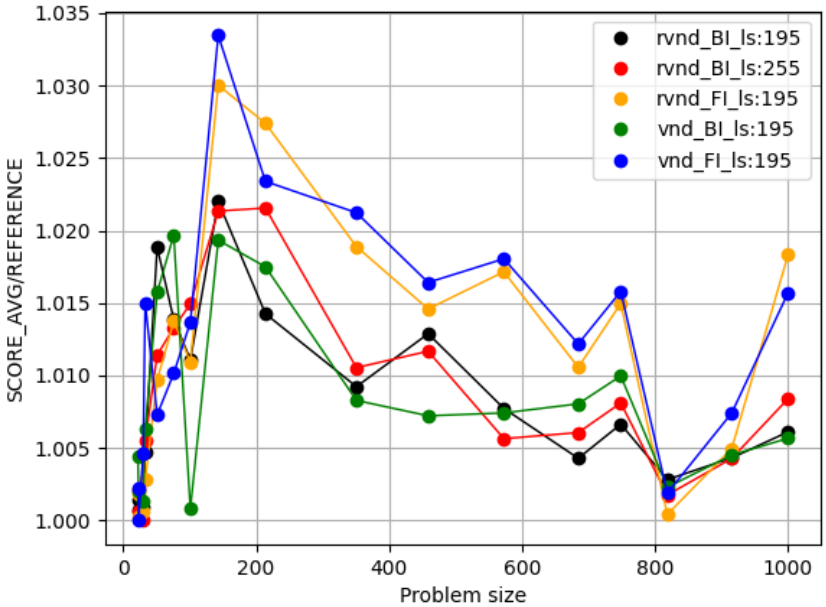
The dataset consists of 4 types of instances, denoted as E, F, M, and X. These EVRP instances were created from already existing CVRP instances by a procedure described in [13], which adds a minimum number of charging stations such that all the customers are reachable from at least one charging station. The E instances are generated from the CVRP benchmark set from [1], the M instances from [2], the F instances from [8] and the X instances from [16]. The E and F instances are small to medium-sized, as they contain between 21 and 134 customers. The M instances are medium-sized (100 to 199 customers), and the X instances are large (143 to 1000 customers). Only instances X and E are addressed in [12]. Some parameters used in [12] instances (e.g., energy consumption rate and maximal energy capacity) slightly differ from the values used in [13]. Here, values from [12] are used when solving the X and E instances.

3.2 Parameters tuning

The proposed GRASP metaheuristic was implemented in C++ and tuned on the E and X instances from [12]. Three settings of the local search were addressed: randomization of the local search (VND or RVND), neighborhood descent strategy (best improvement - BI or first improvement - FI), and selection of the best performing subset of the operators. The individual setups' names encode the parameter settings. For example, setup rvnd_BI_ls:255 stands for RVND, best improvement, and operators subset no. 255 (which is 11111111 in binary representation and corresponds to using all eight operators). As it was not feasible to tune all three parameters simultaneously, the process was split into two phases. First, all 255 possible subsets out the eight operators described in Section 2.4 were tested. In this phase, the remaining two parameters were set to RVND and BI. Second, all four combinations of the two remaining parameters were tested simultaneously, while the already selected subset of operators was fixed.



(a) Best performance



(b) Average performance

Fig. 2: Parameters tuning

In the first phase, the method rvnd_BI_ls:195 performed best, as it yielded the lowest best score most frequently. The operators used in this setup are 2-opt, 1-point, 2-point, 3-point. Interestingly, no or-opt operator is included. The results of the second phase are presented in Figure 2 and in Table 1. Figure 2a displays the best performance of the individual setups, Figure 2b the average performance and Table 1 provides counts of achieving the lowest (=best) score for each setup. The results are plotted relative to the best score achieved by the setup rvnd_BI_ls:195, which serves as a reference. It turns out that rvnd_BI_ls:195 is also best in terms of achieving the lowest best score (9 times out of 17 instances) in the second phase. However, vnd_BI_ls:195 is slightly better in terms of achieving the lowest average score. The first improvement descent strategy is generally rather unsuccessful. Based on these results, the setup rvnd_BI_ls:195 is selected as the final method. As can be seen in Figure 2, the differences among individual setups are minor. When averaged across all instances, the worst setup vnd_FI_ls:195 is worse by 1% in terms of the best score and only by 0.5% in terms of the average score than the reference setup.

GRASP setup	lowest avg - cnt	lowest best - cnt
rvnd_BI_ls:195	4	9
rvnd_BI_ls:255	3	7
rvnd_FI_ls:195	2	6
vnd_FI_ls:195	3	2
vnd_BI_ls:195	5	2

Table 1: Parameters tuning

3.3 Final results

This section documents the performance of the tuned GRASP metaheuristic on all the available instances from the dataset [13]. The results on the E instances are presented in Table 2. The authors of [14] provided fitness values of the best-known solutions for these instances in [12]. These values are shown in the rightmost column. The implemented GRASP metaheuristic found better solutions for 5 out of the 7 instances - the improved values are displayed in bold font in Table 2. The current best-known solution scores are given in the column marked as BKS. The best score obtained by the GRASP metaheuristic is at most by 1.2% worse than the BKS (instance E-n22-k4). On the other hand, the GRASP metaheuristic in some cases improved the previous best score by as much as 6% (instances E-n51-k5 and E-n101-k8).

The results on the X, F and M instances are given in Table 3, 4 and 5 respectively. As no other solution values were known at the time of writing, the presented scores are intended as an initial benchmark for future research.

instance	best	mean±stdev	worst	t_{avg}(s)	BKS	prev. best
E-n22-k4	389.32	389.89±0.41	390.19	0.09	384.68	384.68
E-n23-k3	**571.95**	572.36±0.56	573.13	0.10	**571.95**	573.13
E-n30-k3	512.19	512.67±0.31	512.88	0.13	511.25	511.25
E-n33-k4	**841.08**	845.06±1.56	846.83	0.15	**841.08**	869.89
E-n51-k5	**536.09**	546.21±5.32	562.32	0.30	**536.09**	570.17
E-n76-k7	**701.63**	711.36±5.27	721.21	0.58	**701.63**	723.36
E-n101-k8	**847.47**	856.86±6.90	871.10	0.96	**847.47**	899.88

Table 2: GRASP results on competition E instances

instance	best	mean±stdev	worst	t_{avg}(s)
X-n143-k7	16460.80	16823.00±157.00	17071.90	1.79
X-n214-k11	11575.60	11740.70±80.41	11881.90	4.76
X-n351-k40	27521.20	27775.30±111.99	28019.90	27.26
X-n459-k26	25929.20	26263.30±134.66	26527.40	30.75
X-n573-k30	52584.50	52990.90±246.79	53591.00	52.28
X-n685-k75	72481.60	72792.70±189.53	73206.10	111.70
X-n749-k98	82187.30	82733.40±213.21	83170.40	245.03
X-n819-k171	166500.00	166970.00±211.84	167370.00	492.28
X-n916-k207	345777.00	347269.00±654.93	348764.00	1108.73
X-n1001-k43	77636.20	78111.20±315.31	78914.00	191.70

Table 3: GRASP results on competition X instances

4 Conclusion

This paper addresses the recently formulated Electric Vehicle Routing Problem and presents the GRASP metaheuristic for finding high-quality solutions in a reasonable time. The performance is tested on the recently proposed dataset [13] of CGVRP instances, which is also the first one publicly available. For the instances with best-known solution values, the GRASP metaheuristic proves to be competitive, as it improves the best-known solution for 5 out of 7 instances. The rest of the instances with no previous solution values is solved as well, and the results are presented for future reference. The main strength of the implemented GRASP metaheuristic lies in efficient local search, which is sped up by using constant-time cost update functions. The key component when applying the GRASP to the EVRP is a newly proposed robust repair procedure called Separate Sequential Fixing (SSF).

Concerning future work, extracting problem-specific information will be tested. For example, all of the implemented local search operators do not consider the currently unused AFSs, as they are inspired by methods for TSP-like problems and explore only some permutation-based neighborhood of the current solution. Also, the initial NN construction is only distance-based, and meeting the battery and load constraints is left entirely for the repair procedure. Adopting informed methods for initial construction, such as those discussed in [11], might prove ben-

instance	best	mean±stdev	worst	t_{avg}(s)
F-n49-k4-s4	729.97	731.02±0.88	732.57	0.21
F-n80-k4-s8	247.80	248.00±0.45	249.21	0.49
F-n140-k5-s5	1177.97	1179.61±1.96	1186.70	1.55

Table 4: GRASP results on F instances

instance	best	mean±stdev	worst	t_{avg}(s)
M-n110-k10-s9	829.00	829.29±0.41	830.11	1.04
M-n126-k7-s5	1066.00	1068.05±1.40	1070.10	1.36
M-n163-k12-s12	1068.60	1081.42±8.24	1106.36	2.31
M-n212-k16-s12	1359.55	1377.25±9.82	1395.23	4.45

Table 5: GRASP results on M instances

eficial, especially for the larger instances where the quality of the initial solution is crucial. Besides that, it is important to compare the metaheuristic with other methods addressing similar problem formulations, such as [18] or [17]. These were tested on a dataset that is not publicly available and was not obtained at the time of writing.

Acknowledgements

This work has been supported by the European Union's Horizon 2020 research and innovation program under grant agreement No 688117. The work of David Woller and Viktor Kozák has also been supported by the Grant Agency of the Czech Technical University in Prague, grant No. SGS18/206/OHK3/3T/37.

References

1. Christofides, N., Eilon, S.: An algorithm for the vehicle-dispatching problem. Journal of the Operational Research Society **20**(3), 309–318 (1969). https://doi.org/10.1057/jors.1969.75
2. Christofides, N., Mingozzi, A., Toth, P.: Exact algorithms for the vehicle routing problem, based on spanning tree and shortest path relaxations. Mathematical Programming **20**(1), 255–282 (dec 1981). https://doi.org/10.1007/BF01589353, https://link.springer.com/article/10.1007/BF01589353
3. Duarte, A., Mladenović, N., Sánchez-Oro, J., Todosijević, R.: Variable neighborhood descent. In: Handbook of Heuristics, vol. 1-2, pp. 341–367. Springer International Publishing (aug 2018). https://doi.org/10.1007/978-3-319-07124-4_9, https://doi.org/10.1007/978-3-319-07124-4{_}9
4. Englert, M., Röglin, H., Vöcking, B.: Worst case and probabilistic analysis of the 2-opt algorithm for the TSP. Algorithmica **68**(1), 190–264 (jun 2014). https://doi.org/10.1007/s00453-013-9801-4, https://link.springer.com/article/10.1007/s00453-013-9801-4

5. Erdelic, T., Carić, T., Lalla-Ruiz, E.: A Survey on the Electric Vehicle Routing Problem: Variants and Solution Approaches (2019). https://doi.org/10.1155/2019/5075671
6. Feo, T.A., Resende, M.G.: A probabilistic heuristic for a computationally difficult set covering problem. Operations Research Letters **8**(2), 67–71 (apr 1989). https://doi.org/10.1016/0167-6377(89)90002-3
7. Festa, P., Resende, M.G.: GRASP. In: Handbook of Heuristics, vol. 1-2, pp. 465–488. Springer International Publishing (aug 2018). https://doi.org/10.1007/978-3-319-07124-4_23
8. Fisher, M.L.: Optimal solution of vehicle routing problems using minimum K-trees. Operations Research **42**(4), 626–642 (1994). https://doi.org/10.1287/opre.42.4.626, `https://www.jstor.org/stable/171617`
9. Goncalves, F., S., C., S., R.: Optimization of distribution network using electric vehicles: A VRP problem. Tech. rep., University of Lisbon (2011)
10. Gutin, G., Yeo, A., Zverovich, A.: Traveling salesman should not be greedy: Domination analysis of greedy-type heuristics for the TSP. Discrete Applied Mathematics **117**(1-3), 81–86 (mar 2002). https://doi.org/10.1016/S0166-218X(01)00195-0
11. Kozák, V., Woller, D., Kulich, M.: Initial solution constructors for capacitated green vehicle routing problem. Modelling and Simulation for Autonomous Systems (MESAS) 2020 (2020)
12. Mavrovouniotis, M.: CEC-12 Competition on Electric Vehicle Routing Problem. `https://mavrovouniotis.github.io/EVRPcompetition2020/` (2020)
13. Mavrovouniotis, M., Menelaou, C., Timotheou, S., Ellinas, G.: A Benchmark Test Suite for the Electric Capacitated Vehicle Routing Problem. In: 2020 IEEE Congress on Evolutionary Computation (CEC). pp. 1–8 (2020). https://doi.org/0.1109/CEC48606.2020.9185753
14. Mavrovouniotis, M., Menelaou, C., Timotheou, S., Panayiotou, C., Ellinas, G., Polycarpou, M.: Benchmark Set for the IEEE WCCI-2020 Competition on Evolutionary Computation for the Electric Vehicle Routing Problem. Tech. rep., KIOS Research and Innovation Center of Excellence, Department of Electrical and Computer Engineering, University of Cyprus, Nicosia, Cyprus (2020), `https://mavrovouniotis.github.io/EVRPcompetition2020/TR-EVRP-Competition.pdf`
15. Normasari, N.M.E., Yu, V.F., Bachtiyar, C., Sukoyo: A simulated annealing heuristic for the capacitated green vehicle routing problem. Mathematical Problems in Engineering **2019** (2019). https://doi.org/10.1155/2019/2358258
16. Uchoa, E., Pecin, D., Pessoa, A., Poggi, M., Vidal, T., Subramanian, A.: New benchmark instances for the capacitated vehicle routing problem. European Journal of Operational Research **257** (08 2016). https://doi.org/10.1016/j.ejor.2016.08.012
17. Wang, L., Lu, J.: A memetic algorithm with competition for the capacitated green vehicle routing problem. IEEE/CAA Journal of Automatica Sinica **6**(2), 516–526 (2019). https://doi.org/10.1109/JAS.2019.1911405
18. Zhang, S., Gajpal, Y., Appadoo, S.S.: A meta-heuristic for capacitated green vehicle routing problem. Annals of Operations Research **269**(1-2), 753–771 (oct 2018). https://doi.org/10.1007/s10479-017-2567-3