A hybrid of GRASP and variable neighborhood search with effective post-processing step for the capacitated location-routing problem

Working Paper DPO-2017-03 (version 1, 15.04.2017)

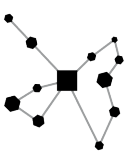
Michael Schneider, Maximilian Löffler and Enrico Bartolini
{schneider|loeffler|bartolini}@dpo.rwth-aachen.de
Deutsche Post Chair – Optimization of Distribution Networks
RWTH Aachen University, Germany

Abstract

Location-routing problems (LRPs) jointly optimize the location of depots and the routing of vehicles. The most studied LRP variant, the capacitated LRP (CLRP), has been addressed by a large number of metaheuristic approaches. These methods often decompose the problem into a location stage to determine a promising depot configuration and a routing stage, where a vehicle routing problem is solved to assess the quality of the previously determined depot configuration. Unfortunately, the CLRP literature does not shed much light on the important question which algorithmic features have the biggest influence on the solution quality and run-time of such heuristics. To solve the CLRP and to provide some insights on the design of successful metaheuristics for this problem, we develop a hybrid algorithm combining (i) a GRASP phase that uses a VND for local improvement in the location stage, and (ii) a VNS in the routing stage.

We find that the suboptimal routing solutions used to assess the quality of the investigated depot configurations in tendency lead to depot configurations with too many open depots. We propose a post-processing step that alleviates this drawback, and we show that this mechanism significantly contributes to the solution quality of our method, enabling it to provide convincing results in comparison to the state-of-the-art methods from the literature.

Keywords: *location-routing problem, GRASP, variable neighborhood search, post-processing*



Deutsche Post
Chair - Optimization of
Distribution Networks

RWTH AACHEN
UNIVERSITY

1 Introduction

In location-routing problems (LRPs), decisions on the location of depots are jointly taken with decisions on the routing of vehicles. Making these types of decisions independently of one another may lead to strongly suboptimal planning results (see Salhi and Rand 1989). Consequently, LRPs have been studied for decades, and the research community has been very active, particularly in the last years. This is witnessed by a large number of surveys on the recent LRP literature that were published in a very short time span (see Lopes et al. 2013, Drexl and Schneider 2014a,b, Prodhon and Prins 2014, Cuda et al. 2015, Albareda-Sambola 2015). For earlier LRP surveys, we refer the reader to Min et al. (1998) and Nagy and Salhi (2007).

The capacitated LRP (CLRP) is the most studied LRP variant. Given a set of potential capacitated depot locations and a set of customers with known demand, the objective of the CLRP is to determine the depots to open and the routes serving the customers from the opened depots in a way that minimizes total costs, which are composed of fixed costs of depots and vehicles and variable routing costs. The CLRP assumes an unlimited, homogeneous fleet of capacitated vehicles at each depot, and depot capacities that limit the total demand served on the routes assigned to each depot. Mathematical models for the CLRP have been provided in numerous papers, see, e.g., Prins et al. (2007). The CLRP is NP-hard because it contains the NP-hard capacitated vehicle-routing problem (VRP). Due to its computational complexity and practical relevance, a substantial effort has been devoted to the development of high-quality heuristic approaches by the research community. Most of the proposed heuristics decompose the problem into a location-allocation stage, in which the facilities to be opened and the assignment of customers to facilities are determined, and a routing stage, in which a VRP is solved for each opened facility. Allocation decisions are frequently also allowed during the routing stage, and in many cases the two stages are solved iteratively in a feedback loop. The most successful heuristic methods for the CLRP use a variety of paradigms: large neighborhood search (Hemmelmayr et al. 2012), simulated annealing (SA, see Yu et al. 2010), ant colony optimization (Ting and Chen 2013), memetic algorithms (Lopes et al. 2016), variable neighborhood search (VNS, see Pirkwieser and Raidl 2010, Escobar et al. 2014), greedy randomized adaptive search procedure (GRASP, see Duhamel et al. 2010, Contardo et al. 2014a) and matheuristic approaches integrating integer linear programming techniques (Prins et al. 2007, Pirkwieser and Raidl 2010, Escobar et al. 2013, Contardo et al. 2014a). For summaries of the individual works, we refer the reader to the surveys of Lopes et al. (2013), Prodhon and Prins (2014), and Drexl and Schneider (2014b).

It has been noted that the quality of a solution strongly depends on the opened facilities (see, e.g. Prins et al. 2006). Therefore, many successful heuristics intensively search the space of potential depot configurations, employing diversification and intensification techniques. GRASP is a frequently used approach in this context (see, e.g., Prins et al. 2006, Duhamel et al. 2010, Contardo et al. 2014a). Apart from this, the CLRP literature does not shed much light on the important question why a certain heuristic performs better than other approaches with a comparable degree of sophistication, or which components of a presented heuristic have the biggest influence on solution quality and run-time.

In this paper, we develop a metaheuristic hybrid combining GRASP for the location phase with a VNS for the routing phase, called GRASP/VNS, to address the CLRP. Instead of a greedy local search, the GRASP phase uses a variable neighborhood descent (VND) to improve the routing solutions given by the randomized construction heuristic. The same VND component is also used as local search within the VNS.

The primary aim of this paper is to provide some insights on how to design a successful (GRASP-based) metaheuristic for the CLRP: we explain several implementation decisions in more detail and study the

effects of the main components of our GRASP/VNS on solution quality and run-time. In the development and testing of our method, we find that determining high-quality routing solutions is not only important for the final solution quality but also for an accurate evaluation of the quality of depot configurations. The suboptimal routing solutions used to assess the quality of the investigated depot configurations in the GRASP phase in tendency lead to depot configurations with too many open depots. To counteract this problem, we design a simple post-processing step that carries out a quick assessment of the feasible configurations with a reduced number of open depots. In numerical studies, we show that this mechanism and the improved routing with the VNS component significantly contribute to the quality of our method. On the commonly used CLRP benchmarks of Tuzun and Burke (1999), Barreto (2004), and Prins et al. (2004), our method provides convincing results in comparison to the state-of-the-art.

The remainder of this paper is structured as follows. We describe our solution method in detail in Section 2. Our numerical studies to investigate the effect of different components of GRASP/VNS on the solution quality and run-time and to compare our method to the state-of-the-art are described in Section 3, followed by a short conclusion in Section 4.

2 GRASP/VNS for the CLRP

We implement a hybrid solution method combining GRASP for the location phase with a VNS for the routing phase. Figure 1 gives a pseudocode overview of our GRASP/VNS.

GRASP/VNS uses the randomized extended Clarke and Wright savings algorithm (RECWA) proposed by Prins et al. (2006) to determine the initial depot configuration and vehicle routes, which are subsequently improved by a greedy local search. We note the following problem of a basic GRASP procedure: Depot configurations are evaluated based on the generally rather low-quality routing solutions returned by the local search. This is likely to lead to wrong decisions when comparing the potential of depot configurations, and a configuration that is dominated by other configurations after a thorough routing phase might be assigned the lowest total cost in this step. This is particularly problematic if the depot configuration is fixed during the routing phase, and thus the best configuration has to be found in the GRASP phase. To counteract this problem, our algorithm uses the following two measures: (i) we replace the greedy local search by a VND that is able to handle solutions that are infeasible with respect to vehicle and/or depot capacities and aims at repairing such solutions as quickly as possible (see Section 2.1), and (ii) we do not only keep one solution of the GRASP phase but store the three solutions with best objective function value and improve the routing of these solutions by means of a *route improvement phase* that carries out a small number of VNS iterations (see Section 2.2) to solve the underlying multi-depot VRP (MDVRP).

We further improve the best solution found so far as follows: We observe a tendency of the GRASP phase to open too many depots because customers cannot be fitted into the suboptimal routes given by the routing solutions found during the GRASP phase. We design a simple *post-processing step* that carries out a fast assessment of the profitability of depot configurations with a reduced number of open depots (see Section 2.3). In this step, the newly generated depot configurations are also improved with a route improvement phase to allow for a fair comparison with the current best solution. As final part of the post-processing step, our algorithm uses an intensive VNS run to further improve the routing of the best solution found until then (see Section 2.3).

{*GRASP phase*}
for n_{grasp} iterations **do**
  $s \leftarrow$ RECWA( )
  $s \leftarrow$ VND(s)
  update list S_{best} storing the three best solutions
end for
{*route improvement phase*}
$S_{best} \leftarrow$ shortVNSRun(S_{best}) {route improvement with short VNS run}
$s^* \leftarrow$ bestOf(S_{best})
{*post-processing step*}
$\Omega \leftarrow$ generate η_{post} feasible depot configurations with one depot less than s^*
for $\omega \in \Omega$ **do**
  $s \leftarrow$ initialSolution(ω) {assign all customers to their closest depot in ω and serve them by dedicated routes}
  $s \leftarrow$ VND(s)
  $s \leftarrow$ shortVNSRun(s)
  if s improves s^* **then**
    $s^* \leftarrow s$
  end if
end for
$s^* \leftarrow$ longVNSRun(s^*) {improve with intensive VNS run}

Figure 1: Overview of the GRASP/VNS algorithm.

2.1 The GRASP phase

The core element of the GRASP phase is the RECWA of Prins et al. (2006), which is repeatedly performed alternating between a diversification and an intensification phase. RECWA first assigns customers to their closest depot with sufficient capacity, serves each customer on a dedicated route from the depot, and closes the facilities with no assigned customers. Subsequently, all feasible mergers of two routes, i.e., mergers that do not violate vehicle or depot capacity restrictions, are evaluated. RECWA evaluates several possibilities of assigning the merged route to different depots: the depot of the first route, the depot of the second route, or any other depot. We store the merge moves with the best savings in a restricted candidate list and randomly select the move to perform from this list with uniform probability. In each iteration of the RECWA, the size l of this list is randomly varied within the interval $[1, l_{max}]$. This procedure is repeated until no feasible merger that leads to a cost reduction can be found.

Subsequently, we improve the routes of the resulting solution by a VND using the neighborhoods defined by the operators in Table 1 in the given order. In each iteration of the VND, the first improving move is executed. The search stops if no improving move can be found. An infeasible solution s is assigned the objective value $F_g(s) = F(s) + M \cdot V_c(s)$, where $F(s)$ denotes the standard objective value, $V_c(s)$ the amount of capacity violation on vehicles and depots, and M is a very large penalty factor. In this way, the VND will always prefer moves that reduce the amount of capacity violation in the solution over moves that improve the routing distance and thus guide the search towards feasible solutions.

Motivated by the insight that the quality of a solution strongly depends on its depot configuration, we carry out the described GRASP procedure for n_{grasp} iterations, alternating between n_{div} diversification and n_{int} intensification iterations as inspired by Prins et al. (2006). Both phases are based on restricting the depots that can be used for the initial customer assignment to a subset J' of the available depots J. In the first iteration of the diversification phase, all depots are available and may potentially be opened. In the next iterations, initially only two depots are available to define J': The first one is iteratively selected so that each depot of J was opened at least once, the second one is randomly drawn from the remaining ones. In this way, the solution space is systematically explored and each depot is used at least once. All customers are

Operator	Description
Split-1	A new route containing a single customer is opened at any open depot.
Relocate-1	A single customer is extracted from a route and inserted into the same route at another position or into any other route.
Exchange-1	Two customers from the same or from different routes are swapped.
Split-2	A new route containing two consecutive customers is opened at any open depot.
Relocate-2	Same as Relocate-1, but two consecutive customers are relocated, preserving their original order.
Exchange-2	Same as Exchange-1, but two consecutive customers are swapped, preserving their original order.
Split-b	A new route containing a customer i and all its predecessors is opened at any open depot.
Split-f	A new route containing a customer i and all its successors is opened at any open depot.
Relocate-b	A customer i and all its predecessors are moved to the same or a different route at any position.
Relocate-f	A customer i and all its successors are moved to the same or a different route at any position.
2-Opt*	Two routes originating at the same depot are reconnected in such a way that the first part of the first route is connected with the second part of the second route and vice versa.
Exchange-b	A customer i and all its predecessors are swapped with a customer j and all its predecessors. Customers i and j are on different routes at different depots.
Exchange-f	A customer i and all its successors are swapped with a customer j and all its successors. Customers i and j are on different routes at different depots.

Table 1: Neighborhood operators of the VND.

sequentially assigned to their closest depot in J'. If a customer cannot be assigned because of a capacity violation, the depot closest to the customer is opened. Subsequently, mergers are performed as described above, i.e., not only depots in J' but all depots are considered as starting and ending location for the merged routes. In the intensification phase, only the depots in J' that are open in the best solution of the preceding diversification phase may be used. Here, not only the assignment of customers is restricted to this subset but also the merge operations during the RECWA.

2.2 Route improvement phase using variable neighborhood search

In general, the vehicle routes generated by the GRASP are of mediocre quality. We apply a short VNS run to further improve the routing by solving the underlying MDVRP of the current CLRP solution. VNS, originally proposed by Mladenović and Hansen (1997), searches in increasingly large neighborhoods in order to escape from local optima. Given a set of often nested, pre-selected neighborhoods for the so-called shaking step and a generally different neighborhood for the local search: Starting from the current solution x, VNS randomly generates a new solution x' using the first shaking neighborhood and from there a local search is performed. If the thus obtained solution x'' has lower quality than the current solution x, the next shaking neighborhood is chosen to repeat the procedure. If a better solution is found or all available shaking neighborhoods have been explored, the search is restarted with the first shaking neighborhood.

The VND described above is used as local search component within our VNS. We define the shaking neighborhoods by means of a cyclic-exchange operator, which moves customer sequences between routes in a cyclic way (Ibaraki et al. 2005). Figure 2 depicts an example with four routes $i = 1, 2, 3, 4$, in which the customer sequences from v_i to w_i are exchanged between the routes. We distinguish 18 neighborhoods generated by this operator as shown in Table 2.

The routes that are involved in a cyclic exchange are determined in the following way. First, a base route is randomly drawn from the set of all routes. The centroid of a route is given by the average Cartesian coordinates of all customers on the route and of the depot that the route originates from. Second, all routes

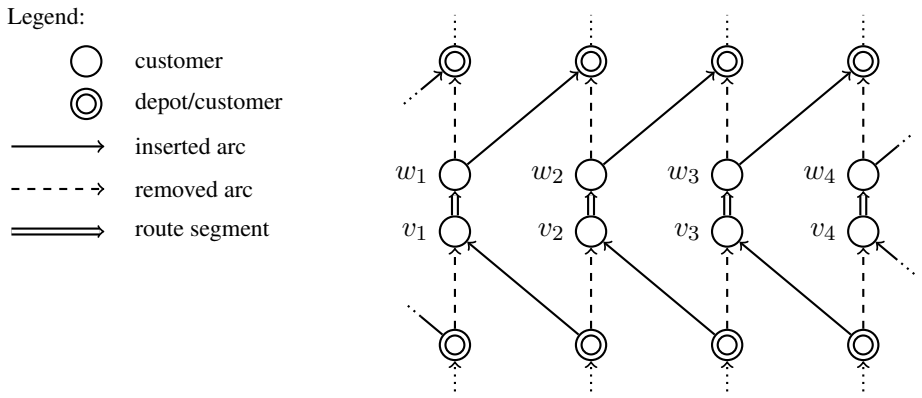


Figure 2: The cyclic-exchange operator with four routes.

Neighborhood	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
# Routes involved	3	3	3	3	3	3	4	4	4	4	4	4	5	5	5	5	5	5
Max. sequence length	1	2	3	4	5	6	1	2	3	4	5	6	1	2	3	4	5	6

Table 2: Overview of neighborhoods used in the VNS.

are sorted in increasing order of the distance between their centroid and the centroid of the base route, and the number of routes to be involved in the cyclic exchange move are taken from the beginning of the list. The number of customers to be exchanged is randomly drawn from the interval from 1 up to the given maximal sequence length. One draw is made for each individual route.

Note that the resulting solution may be infeasible with respect to capacity constraints. This solution is improved (and repaired in case it is infeasible) by applying the VND described above. The short VNS run in the route improvement phase terminates after n_{short} iterations without improvement.

2.3 Post-processing step

According to our computational experience, the depot configurations determined by the GRASP are not optimal in many cases. As described above, the GRASP tends to open too many depots because customers cannot be fitted into the suboptimal routes of the routing solutions determined during the GRASP phase. To remedy this drawback, we use a simple post-processing step that systematically evaluates depot configurations with a reduced number of depots. Starting from the best depot configuration found by the GRASP and route improvement phase, we determine all feasible configurations with exactly one depot less, i.e., those for which the total capacity of open depots is sufficient to serve the total customer demand.

We randomly select one depot configuration out of the resulting list and assign all customers to their closest depot, from which each customer is initially served by a dedicated route. The resulting solution may be infeasible with respect to depot capacity. First, it is improved (or repaired in case the solution is infeasible) by one run of VND described in Section 2.1. Next, the routing is further improved with a short VNS run that stops after n_{short} iterations without improvement. This allows for a fair comparison of the resulting solution with the current best solution. If the resulting solution improves upon the current best solution, it becomes the new best solution, otherwise it is discarded. The described procedure is repeated until we have examined η_{post} depot configurations. In case there are less than η_{post} feasible depot configurations with one depot less than in the best solution determined by GRASP, we successively increase the number of open depots by one

and randomly generate further configurations until we have investigated η_{post} depot configurations.

As final part of our post-processing step, we try to further improve the routing of the best solution found during the post-processing step. To this end, we carry out a long VNS run that terminates after n_{long} iterations without improvement.

3 Computational studies

This section presents our numerical experiments to (i) investigate the effect of the core components of our algorithm on run-time and solution quality (Section 3.3), and (ii) compare GRASP/VNS to the state-of-art (Section 3.4). The utilized benchmark sets, proposed by Barreto (2004), Tuzun and Burke (1999), and Prins et al. (2004), are introduced in Section 3.1. GRASP/VNS is implemented in C++, and we ran all experiments on a single core of an Intel Xeon E5-2687W CPU clocked at 3.1 GHz with 128GB RAM, running openSUSE Leap 42.2. The memory consumption stayed below 128MB at all times. The parameter setting of the algorithm is described in Section 3.2.

3.1 Benchmark instances

The benchmark set proposed by Barreto (2004), called Barreto set in the following, consists of 13 instances with up to 150 customers and 10 depots, of which most are adapted from classical VRP instances. Vehicle capacities are generally limited, and in most of the instances depot capacities are also restricted. Tuzun and Burke (1999) present a set of 36 instances with capacitated vehicles and without depot capacities, called Tuzun-Burke set. The instances go from 100 customers and 10 depots up to 200 customers and 20 depots. Instances with customer clusters of varying density and instances with uniformly distributed customers are contained in the set. The benchmark of Prins et al. (2004), called Prodhon set, consists of 30 instances ranging from 20 customers and 5 depots to instances with up to 200 customers and 10 depots. All instances are restricted in terms of vehicle and depot capacity, and the set comprises instances with two or three customer clusters as well as instances with uniformly distributed customers. For the instances in the Barreto and Tuzun-Burke set, the Euclidean distance between two vertices denotes the travel costs, whereas in the Prodhon instances, the Euclidean distance is multiplied by 100 and rounded up to the next integer in order to have integer travel costs.

3.2 Parameter setting

Based on a randomly selected subset of 15 instances from the three benchmark sets described above, we performed a preliminary tuning to find a decent parameter setting, which is shown in Table 3. We set the number of GRASP iterations to $n_{grasp} = 100$ because this provides a good tradeoff between solution quality—in particular robustness of the algorithm—and speed. Higher numbers of iterations did not significantly improve the average solution quality but led to clearly higher run-times. Lower values below 50 iterations resulted in a rather unstable algorithm, probably caused by the substantial randomization of the GRASP. In the $n_{grasp} = 100$ iterations, we iterate between $n_{div} = 7$ diversification and $n_{int} = 3$ intensification steps, in which we set the length of the restricted candidate list to $l_{max} = 7$ as proposed by Duhamel et al. (2010).

The number of iterations of the post-processing step is particularly relevant if there are no capacity restrictions on the depots, and all potential depot configurations are feasible and thus candidates for evaluation. Here, we achieved the best tradeoff between solution quality and run-time by stopping the mechanism after

GRASP				Stopping criteria		
l_{max}	n_{grasp}	n_{div}	n_{int}	η_{post}	n_{short}	n_{long}
7	100	7	3	50	20	200

Table 3: Parameter setting of GRASP/VNS.

evaluating $\eta_{post} = 50$ depot configurations. The route improvement phase performed after the GRASP and after the post-processing step stops after $n_{short} = 20$ iterations without improvement. The long VNS run in the final intensive routing phase stops after $n_{long} = 200$ iterations without improvement.

3.3 Effect of algorithm components

Our GRASP/VNS comprises three essential components: the GRASP phase, the VNS routing algorithm, and the post-processing step. In this section, we evaluate in more detail how each component contributes to the solution quality and run-time of our algorithm in order to be able to derive insights on the design of successful metaheuristics for the CLRP. To this end, Table 4 presents the results on the three benchmark sets described in Section 3.1 with the following configurations of our algorithm:

G: GRASP without VNS routing or post-processing,

G+VNS: GRASP plus VNS routing but without the post-processing step,

G+VNS+PP: the final GRASP/VNS including GRASP, VNS routing, and the post-processing step.

Due to the random components of the GRASP and the VNS, we solve each instance 5 times with the parameter setting described in Section 3.2, and we report the percentage gap of the best solution found in these runs to the best-known solution (BKS) as given in Escobar et al. (2014) and Contardo et al. (2014b) (Δ_f), and the average run-time (t) in seconds. Table 4 presents an aggregate view and reports averages for each separate instance set and over all three instance sets. Finally, the results of G+VNS+PP, i.e., of the final GRASP/VNS, on an instance basis are provided in Tables 6–8 in Appendix A.

	G		G+VNS		G+VNS+PP	
Benchmark	$\Delta_f(\%)$	$t(s)$	$\Delta_f(\%)$	$t(s)$	$\Delta_f(\%)$	$t(s)$
Tuzun	2.64	226	1.29	230	0.61	377
Prodhon	2.74	72	2.10	75	0.11	158
Barreto	1.01	22	0.26	23	0.09	44
Average	**2.41**	**134**	**1.43**	**137**	**0.34**	**239**

Table 4: Comparison of results for the three algorithm configurations G, G+VNS, and G+VNS+PP on the Tuzun-Burke, Prodhon, and Barreto benchmark sets. We report the gap of the best solution found by each method to the BKS as reported in Escobar et al. (2014) and Contardo et al. (2014b) (Δ_f), and the average run-time (t) in seconds. Results are given as averages over the three benchmark sets.

The results demonstrate the strong positive impact of our VNS routing algorithm on the obtained solution quality: the VNS is able to reduce the gap from 2.41% for G to 1.43% for G+VNS with only a negligible increase in run-time. This shows that a fast and decent routing component has a strong effect on solution quality. The routing method does not only account for the final routing solution but influences the entire

solution process because it strongly influences the assessment of the quality of depot configurations during the search.

Moreover, we observe that the post-processing step is able to further reduce the gap from 1.43% to 0.34% for G+VNS+PP. This solution improvement is remarkable, given the simple nature of our post-processing step and the fact that reducing the gap to the BKS becomes ever more difficult, the closer one gets to the BKS. Given this setting, the increase in run-time of approximately 75% seems acceptable. As explained above, the suboptimal routing solutions used to assess the depot configurations lead to a bias towards depot configurations that have too many depots opened. However, as the results show, this shortcoming can be effectively counteracted by the proposed post-processing step.

3.4 Comparison to the state-of-the-art

In this section, we compare the solution quality and run-time of GRASP/VNS to the most effective heuristic methods for the CLRP from the literature on the Tuzun-Burke, Prodhon, and Barreto benchmark. In Table 5, for each method, the average gap to the BKS (taken from Escobar et al. (2014) and Contardo et al. (2014b)), and the average run-time is reported for each separate instance set and over all three instance sets. Results are provided for the following comparison methods (the way in which the respective authors report results and run-times is explained in the following):

- **DLPP** (Duhamel, Lacomme, Prins, and Prodhon 2010) give the best solution obtained in 5 runs and the time of the best run.
- **YLLT** (Yu, Lin, Lee, and Ting 2010), **ELT** (Escobar, Linfati, and Toth 2013), and **ELBT** (Escobar, Linfati, Baldoquin, and Toth 2014) provide results of a single run.
- For **HCC** (Hemmelmayr, Cordeau, and Crainic 2012), the table reports the best result of 5 runs and the total run-time for two different numbers of total iterations (500K, 5000K).
- For **CCG** (Contardo, Cordeau, and Gendron 2014a), **TC** (Ting and Chen 2013), **LFS** (Lopes et al. 2016), and GRASP/VNS, the table shows the best of 10 runs and the average run-time per run.

Furthermore, the table reports the processor used and the Passmark score for a single core of the respective processor (see www.passmark.com). Although an entirely precise comparison of run-times is not possible due to different operating systems and programming languages, we use this information to translate all run-times into a common time measure based on the CPU we used for testing our algorithm. In addition, the run-times of the methods are multiplied by the number of runs on which the reported solution quality is based.

The performance of GRASP/VNS is quite convincing in comparison to the state-of-the-art. Considering the average over all three benchmark sets, none of the comparison methods is able to dominate our method, i.e., the comparison methods are either faster or achieve a better solution quality, but not both. On the other hand, GRASP/VNS is able to dominate DLPP and the recently proposed LFS-Hybrid GA+. When comparing the results of GRASP/VNS to HCC-500K, the strongest competitor with regard to dominance, it can be noted that HCC-500K achieves only slightly inferior solution quality but within clearly shorter run-times. However, considering the benchmarks separately, we find that HCC-500K is notably superior on the non-capacitated instances of Tuzun-Burke, while GRASP/VNS provides significantly better solution quality on the capacitated instances of Prodhon. Finally, we find three new best-known solutions, one on the Tuzun-Burke set and two on the Prodhon set, see Tables 6 and 7 for details.

	DLPP		YLLT		HCC–500K		HCC-5000K		ELT	
Benchmark	$\Delta_f(\%)$	$t(s)$	$\Delta_f(\%)$	$t(s)$	$\Delta_f(\%)$	$t(s)$	$\Delta_f(\%)$	$t(s)$	$\Delta_f(\%)$	$t(s)$
Tuzun-Burke	1.24	607	1.44	826	0.38	166	0.24	1621	1.09	392
Prodhon	1.09	258	0.44	422	0.42	90	0.33	844	0.55	176
Barreto	0.03	188	0.25	161	0.11	35	0.01	354	0.74	105
Average	0.99	405	0.86	564	0.35	116	0.24	1117	0.83	263
Processor type	Core 2 Quad		Core 2 Quad		Opteron 275		Opteron 275		Core 2 Duo	
Processor speed	2.83 GHz		2.66 GHz		2.20 GHz		2.20 GHz		2.00 GHz	
Passmark score	1200		1135		685		685		776	
Corrected time	261×5		344		43×5		411×5		110	

	CCG		TC		ELBT		LFS-Hybrid GA		LFS-Hybrid GA+		GRASP/VNS	
Benchmark	$\Delta_f(\%)$	$t(s)$	$\Delta_f(\%)$	$t(s)$	$\Delta_f(\%)$	$t(s)$	$\Delta_f(\%)$	$t(s)$	$\Delta_f(\%)$	$t(s)$	$\Delta_f(\%)$	$t(s)$
Tuzun-Burke	0.12	2590	1.18	202	0.71	201	0.81	86	0.65	364	0.61	377
Prodhon	0.10	1163	0.38	191	0.35	91	0.35	73	0.30	199	0.11	158
Barreto	0.15	279	0.07	49	0.62	53	0.04	19	-0.01	93	0.09	44
Average	0.12	1685	0.70	173	0.56	141	0.51	70	0.41	257	0.34	239
Processor type	Xeon E5462		XP 2500+		Core 2 Duo		i7-4790		i7-4790		Xeon E5-2687W	
Processor speed	3.00 GHz		1.83 GHz		2.00 GHz		3.60 GHz		3.60 GHz		3.10 GHz	
Passmark score	1219		546		776		2290		2290		1861	
Corrected time	1104×10		51×10		56		86×5		316×5		239×5	

Table 5: Comparison of GRASP/VNS to the state-of-the-art. For each method, we report the average gap of the best solution found by each method to the BKS as reported in Escobar et al. (2014) and Contardo et al. (2014b) (Δ_f), and the run-time (t) in seconds on the Tuzun-Burke, Prodhon, and Barreto benchmark sets. In addition, we report the benchmarking environment used for each method and a common time measure based on the passmark score of the processor used in the testing.

4 Conclusion

In this paper, we propose a hybrid of GRASP and VNS to solve the CLRP. In the design and experiments with our method, we find that a location phase using suboptimal routing solutions to assess the quality of depot configurations is likely to generate solutions with too many open depots. We observe that, in this case, a simple post-processing step is able to significantly increase the solution quality of the method without strongly increasing the run-time. Finally, our GRASP/VNS provides convincing solution quality on the standard benchmark instances from the literature.

References

M. Albareda-Sambola. Location-routing and location-arc routing. In G. Laporte, S. Nickel, F. Saldanha da Gama, and M. Albareda-Sambola, editors, *Location Science*, chapter 15, pages 399–418. Springer, 2015.

S. Barreto. *Análise e Modelização de Problemas de localização-distribuição [Analysis and modelling of location-routing problems]*. PhD thesis, University of Aveiro, Campus Universitário de Santiago, 3810-193 Aveiro, Portugal, 2004.

C. Contardo, J.-F. Cordeau, and B. Gendron. A GRASP + ILP-based metaheuristic for the capacitated location-routing problem. *Journal of Heuristics*, 20:1–38, 2014a.

C. Contardo, J.-F. Cordeau, and B. Gendron. An exact algorithm based on cut-and-column generation for the capacitated location-routing problem. *INFORMS Journal on Computing*, 26(1):88–102, 2014b.

R. Cuda, G. Guastaroba, and M. G. Speranza. A survey on two-echelon routing problems. *Computers & Operations Research*, 55:185–199, 2015.

M. Drexl and M. Schneider. A survey of variants and extensions of the location-routing problem. *European Journal of Operational Research*, 241(2):283–308, 2014a.

M. Drexl and M. Schneider. A survey of the standard location-routing problem. Working Paper LPIS-03/2014, Logistics Planning and Information Systems, TU Darmstadt, Germany, 2014b.

C. Duhamel, P. Lacomme, C. Prins, and C. Prodhon. A GRASP×ELS approach for the capacitated location routing problem. *Computers & Operations Research*, 37(11):1912–1923, 2010.

J. Escobar, R. Linfati, and P. Toth. A two-phase hybrid heuristic algorithm for the capacitated location-routing problem. *Computers & Operations Research*, 40(1):70–79, 2013.

J. W. Escobar, R. Linfati, M. G. Baldoquin, and P. Toth. A granular variable tabu neighborhood search for the capacitated location-routing problem. *Transportation Research Part B: Methodological*, 67:344–356, 2014.

V. Hemmelmayr, J.-F. Cordeau, and T. G. Crainic. An adaptive large neighborhood search heuristic for two-echelon vehicle routing problems arising in city logistics. *Computers & Operations Research*, 39(12):3215–3228, 2012.

T. Ibaraki, S. Imahori, M. Kubo, T. Matsuda, T. Uno, and M. Yagiura. Effective local search algorithms for routing and scheduling problems with general time-window constraints. *Transportation Science*, 39(2):206–232, 2005.

R. B. Lopes, C. Ferreira, B. S. Santos, and S. Barreto. A taxonomical analysis, current methods and objectives on location-routing problems. *International Transactions in Operational Research*, 20(6):795–822, 2013.

R. B. Lopes, C. Ferreira, and B. S. Santos. A simple and effective evolutionary algorithm for the capacitated location-routing problem. *Computers & Operations Research*, 70:155–162, 2016.

H. Min, V. Jayaraman, and R. Srivastava. Combined location-routing problems: A synthesis and future research directions. *European Journal of Operational Research*, 108(1):1–15, 1998.

N. Mladenović and P. Hansen. Variable neighborhood search. *Computers & Operations Research*, 24(11):1097–1100, 1997.

G. Nagy and S. Salhi. Location-routing: Issues, models and methods. *European Journal of Operational Research*, 177 (2):649–672, 2007.

S. Pirkwieser and G. Raidl. Variable neighborhood search coupled with ILP-based very large neighborhood searches for the (periodic) location-routing problem. In M. Blesa, C. Blum, G. Raidl, A. Roli, and M. Sampels, editors, *Hybrid Metaheuristics*, volume 6373 of *Lecture Notes in Computer Science*, pages 174–189. Springer, 2010.

C. Prins, C. Prodhon, and R. Wolfler Calvo. Nouveaux algorithmes pour le problème de localisation et routage sous contraintes de capacité. In A. Dolgui and S. Dauzère-Pérèz, editors, *MOSIM' 04*, volume 2, pages 1115–1122, 2004.

C. Prins, C. Prodhon, and R. Wolfler Calvo. Solving the capacitated location-routing problem by a GRASP complemented by a learning process and a path relinking. *4OR – A Quarterly Journal of Operations Research*, 4(3): 221–238, 2006.

C. Prins, C. Prodhon, A. Ruiz, P. Soriano, and R. Wolfler Calvo. Solving the capacitated location-routing problem by a cooperative Lagrangean relaxation-granular tabu search heuristic. *Transportation Science*, 41(4):470–483, 2007.

C. Prodhon and C. Prins. A survey of recent research on location-routing problems. *European Journal of Operational Research*, 238(1):1–17, 2014.

S. Salhi and G. Rand. The effect of ignoring routes when locating depots. *European Journal of Operational Research*, 39(2):150–156, 1989.

C.-J. Ting and C.-H. Chen. A multiple ant colony optimization algorithm for the capacitated location routing problem. *International Journal of Production Economics*, 141(1):34–44, 2013.

D. Tuzun and L. I. Burke. A two-phase tabu search approach to the location routing problem. *European Journal of Operational Research*, 116(1):87–99, 1999.

V. Yu, S.-W. Lin, W. Lee, and C.-J. Ting. A simulated annealing heuristic for the capacitated location routing problem. *Computers & Industrial Engineering*, 58(2):288–299, 2010.

A Additional computational results

We present the detailed results of GRASP/VNS on the Tuzun-Burke (Table 6), Prodhon (Table 7) and Barreto (Table 8) instance sets. For each instance, we report the BKS as given in Escobar et al. (2014) and Contardo et al. (2014b), the result of the best of five runs of GRASP/VNS as absolute objective function value and as gap to the BKS, and the average run-time in seconds. New BKS found by our algorithm are indicated in bold.

		GRASP/VNS		
Instance	BKS	f	$\Delta_f(\%)$	$t(s)$
111112	1467.68	1479.21	0.79	72
111212	1394.80	1396.46	0.12	74
112112	1167.16	1167.16	0.00	80
112212	791.66	791.70	0.01	79
113112	1238.24	1239.43	0.10	84
113212	902.26	902.26	0.00	75
111122	1449.20	1464.92	1.08	105
111222	1432.29	1432.93	0.04	103
112122	1102.24	1102.39	0.01	113
112222	728.30	728.30	0.00	110
113122	1245.30	1245.31	0.00	111
113222	1018.29	1018.29	0.00	110
121112	2243.40	2270.79	1.22	588
121212	2209.30	2211.10	0.08	580
122112	2073.73	**2071.42**	**-0.11**	648
122212	1453.18	1454.65	0.10	656
123112	1954.70	1990.46	1.83	625
123212	1762.03	1766.78	0.27	608
121122	2138.40	2195.69	2.68	826
121222	2225.10	2262.61	1.69	836
122122	1692.17	1762.35	4.15	916
122222	1082.46	1084.10	0.15	809
123122	1926.64	1947.25	1.07	841
123222	1390.87	1392.19	0.10	815
131112	1896.98	1913.49	0.87	247
131212	1960.23	1965.40	0.26	263
132112	1443.32	1445.08	0.12	273
132212	1204.42	1205.74	0.11	254
133112	1694.18	1704.15	0.59	263
133212	1198.28	1202.17	0.32	276
131122	1823.53	1847.01	1.29	342
131222	1792.80	1792.77	0.00	332
132122	1434.63	1447.89	0.92	397
132222	931.28	931.49	0.02	358
133122	1392.01	1417.52	1.83	351
133222	1151.80	1156.61	0.42	354
Average			**0.61**	**377**

Table 6: Detailed results of GRASP/VNS on the Tuzun-Burke instance set.

		GRASP/VNS		
Instance	BKS	f	$\Delta_f(\%)$	$t(s)$
20-5-1a	54793	54793	0.00	1
20-5-1b	39104	39104	0.00	1
20-5-2a	48908	48908	0.00	1
20-5-2b	37542	37542	0.00	1
50-5-1a	90111	90111	0.00	10
50-5-1b	63242	63242	0.00	10
50-5-2a	88298	88298	0.00	10
50-5-2b	67308	67308	0.00	9
50-5-2bis	84055	84055	0.00	9
50-5-2bbis	51822	51851	0.06	10
50-5-3a	86203	86203	0.00	10
50-5-3b	61830	61830	0.00	9
100-5-1a	274814	276016	0.44	76
100-5-1b	213568	214198	0.29	68
100-5-2a	193671	193671	0.00	70
100-5-2b	157095	157173	0.05	58
100-5-3a	200079	200079	0.00	63
100-5-3b	152441	152466	0.02	62
100-10-1a	287864	288195	0.11	89
100-10-1b	231739	232782	0.45	73
100-10-2a	243590	243819	0.09	87
100-10-2b	203988	203988	0.00	75
100-10-3a	250882	252890	0.80	85
100-10-3b	204597	**204380**	**-0.11**	78
200-10-1a	475327	478919	0.76	648
200-10-1b	377227	378133	0.24	571
200-10-2a	449006	449310	0.07	634
200-10-2b	374435	375000	0.15	569
200-10-3a	469433	469497	0.01	744
200-10-3b	362827	**362770**	**-0.02**	611
Average			**0.11**	**158**

Table 7: Detailed results of GRASP/VNS on the Prodhon instance set.

		GRASP/VNS		
Instance	BKS	f	$\Delta_f(\%)$	$t(s)$
Christofides69-50x5	565.6	565.6	0.00	8
Christofides69-75x10	848.9	848.9	0.00	35
Christofides69-100x10	833.4	837.2	0.46	72
Daskin95-88x8	355.8	355.8	0.00	45
Daskin95-150x10	43952.3	44182.8	0.52	234
Gaskell67-21x5	424.9	424.9	0.00	1
Gaskell67-22x5	585.1	585.1	0.00	1
Gaskell67-29x5	512.1	512.1	0.00	2
Gaskell67-32x5-1	562.2	562.2	0.00	3
Gaskell67-32x5-2	504.3	504.3	0.00	2
Gaskell67-36x5	460.4	460.4	0.00	3
Min92-27x5	3062.0	3062.0	0.00	2
Min92-134x8	5709.0	5719.3	0.18	168
Average			**0.09**	**44**

Table 8: Detailed results of GRASP/VNS on the Barreto instance set.