DEPARTMENT OF ENGINEERING MANAGEMENT

Efficiently solving location routing problems using a vehicle routing heuristic and iterative filtering

Florian Arnold & Kenneth Sörensen

UNIVERSITY OF ANTWERP
Faculty of Applied Economics
City Campus
Prinsstraat 13, B.226
B-2000 Antwerp
Tel. +32 (0)3 265 40 32
Fax +32 (0)3 265 47 99
www.uantwerpen.be

FACULTY OF APPLIED ECONOMICS

DEPARTMENT OF ENGINEERING MANAGEMENT

Efficiently solving location routing problems using a vehicle routing heuristic iterative filtering

Florian Arnold & Kenneth Sörensen

RESEARCH PAPER 2018-010
JUNE 2018

University of Antwerp, City Campus, Prinsstraat 13, B-2000 Antwerp, Belgium
Research Administration – room B.226
phone: (32) 3 265 40 32
fax: (32) 3 265 47 99
e-mail: joeri.nys@uantwerpen.be

D/2018/1169/010

Efficiently solving location routing problems using a vehicle routing heuristic and iterative filtering

Florian Arnold [1a], Kenneth Sörensen[a]

[a] *University of Antwerp, Departement of Engineering Management, ANT/OR - Operations Research Group*
[b] *corresponding author. Email: florian.arnold@uantwerpen.be*

Abstract

The Location Routing Problem (LRP) unites two important challenges in the design of distribution systems. On the one hand, the delivery of goods to customers needs to be planned as effectively as possible, and on the other hand, the location of depots from where these deliveries are executed has to be determined carefully. In the last years many heuristic approaches have been proposed to tackle LRPs. Usually, however, the computation of excellent solutions comes at the cost of an intricate algorithmic design. In this paper we introduce an efficient heuristic for LRPs that is almost entirely based on a heuristic to solve routing problems. We estimate an upper bound for the number of open depots, and iteratively apply the routing heuristic on each remaining configuration of open locations. Despite its simple design, the heuristic competes with the best results in literature, and can also be readily adapted to solve problems of very large scale.

Keywords: vehicle routing problem, heuristics, location routing problem, large scale problem

1. Introduction

The Location Routing Problem (LRP) is a well-established combinatorial optimization problem that combines two important decisions in the design of a supply chain: the decision where to open facilities and the decision how to set up the distribution from facilities to customers. In practice, both decisions are usually taken for different time horizons. While the opening of facilities affects

[1] corresponding author. Email: florian.arnold@uantwerpen.be

the long-term, the distribution, also called routing, is mostly planned on an operational day-by-day basis. Despite the different planning horizons, Salhi and Rand (1989) have shown that the incorporation of distribution planning into a facility location decision can significantly improve the overall costs of the supply chain, and thereby they established a practical motivation for solving LRPs.

The LRP is a generalization of standard routing problems like the Vehicle Routing Problem (VRP) or the Multi-Depot Vehicle Routing Problem (MDVRP) and the facility location problem (FLP). Given a set of customers with known demand and a set of potential facility locations, one has to to determine how many and which facilities to open (as in a FLP), as well as to plan the delivery routes from the open facilities to the customers (as in VRPs and MDVRPs). Hereby, the demand of all customers has to be satisfied, and the capacity limits per route have to be met. An example of an LRP solution is given in Figure 1, and a concise overview about the recent progress in the domain of LRPs can be found in Prodhon and Prins (2014) and in Schneider and Drexl (2017). For the purpose of a uniform terminology, we will use the word *depot* (as in VRPs) to denote facilities in the following.

In the last years a number of effective algorithms have been proposed to tackle LRPs. To manage the problem complexity of the LRP, most successful algorithms consist of different stages and combine heuristic components with integer linear program formulations (ILPs). The improvement of the routing is usually carried out by heuristics, whereas the decisions which depots to open is taken by ILPs. One of the most efficient algorithms with such a design has been proposed by Escobar et al. (2013) and improved in Escobar et al. (2014). Initially, the customers are divided into clusters and an ILP assigns customer clusters to depots such that the routing from the open depots is minimized. Afterwards, the routing is improved with various heuristic techniques. Prins et al. (2007) propose an algorithm which exchanges information between these two stages. The FLP is solved by a Lagrangean relaxation technique, followed by a granular tabu search to solve the corresponding MDVRP. A similar design is proposed in Harks et al. (2013) to solve very large scale LRP instances. In a first stage a solution for the corresponding FLP is approximated using minimal spanning trees, and in a second stage the delivery routes are constructed and improved. In contrast to that, Tuzun and Burke (1999) iterate between the improvement of the location and routing decisions, rather than solving them one after another. Contardo et al. (2014) suggest yet another heuristic design by optimising locations and routing simultaneously. A GRASP technique combined with local search is used to construct a set of promising solutions, which are then iteratively improved with a

mix of ILP-based techniques and destroy-and-repair techniques. The most successful of these heuristics are usually complex, both in terms of their design and in the number of parameters. Even though they are able to find good solutions in a relatively short time, simplicity and flexibility are also valuable properties of a heuristic (Cordeau et al., 2002), especially when it comes to practical usability.

In this paper, we demonstrate that it is possible to design an LRP heuristic with a relatively simple design that is able to compete with the best results in literature. More concretely we provide the following contributions to the research on LRPs:

- We estimate an upper bound for the number of open depots, which drastically reduces the search space.
- We demonstrate how to use an MDVRP heuristic to iteratively evaluate promising configurations of open depots.
- We show that the heuristic is the first which is able to solve LRPs of different magnitudes, and can even tackle instances with 10,000 customers and 1,000 potential depots.

We demonstrate how LRPs can be efficiently decomposed into a location and a routing component in Section 2. In Section 3 we use these insights to design a simple filtering heuristic which can also be extended to other problem variants. The performance of the heuristic is validated in Section 4 on popular benchmark instances. We conclude with a brief summary of our findings in Section 5. An executable Java package that contains the heuristic will be made available at `http://antor.uantwerpen.be/LRP`.

2. Decomposition of LRPs

The objective of Location Routing Problems (LRP) is to open a certain number of depots at different possible locations from which customers are delivered, such that the joint cost of opening the depots and distribution from the depots to customers are minimized. More formally, a candidate set $C = \{f_1, f_2, \ldots, f_F\}$ of potential depot locations is given, and each depot is annotated with a cost $c_o(f_i)$ that is realized if the respective depot is opened. From all open depots a set of N customers with pre-defined locations and demand has to be delivered on routes with vehicles. Each customer has to be visited once, and each vehicle has a maximum capacity so that usually multiple routes have to be planned. In the

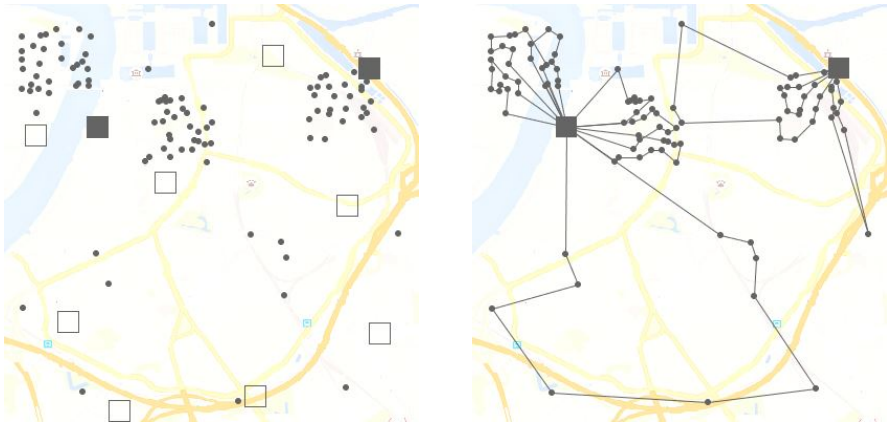

Figure 1: *Decomposition of an LRP. We first generate location options (left), and evaluate each location option by solving the respective MDVRP (right). The example is taken from instance '112112' by Tuzun and Burke (1999).*

standard problem formulation, the capacity of the depots is unconstrained and all customers could be supplied from a single depot. Below, we also consider a problem variant in which the capacity of depots is limited. For a more elaborate overview of the LRP and its variants we refer to Prodhon and Prins (2014).

The optimization task is to determine a set $O \subseteq C$ of open depots, as well as a routing solution from the open depots to satisfy the demand of the customers, that result in minimal overall costs. In the following, we will denote a set of open depots $O \subseteq C$ as *location option*. Let R_O be the routing costs (in most benchmark instances R_O is defined as the sum of the distance traveled on the delivery routes and a fixed costs per route), then the costs of a location option O can be expressed by $c_{LRP}(O) = \sum_{f_i \in O} c_o(f_i) + R_O$.

If the set of open depots is given, then the first term in the objective function can be neglected, and the problem reduces to computing delivery routes with minimal costs. The computation of optimal delivery routes, or Vehicle Routing Problem (VRP), is one of the most-studied problems in the field of Operations Research. The standard variant of the VRP only considers the delivery from one depot, and could solve the routing in the case of $|O| = 1$. For $|O| > 1$ we have to solve a Multi-Depot Vehicle Routing Problem (MDVRP).

Since the VRP and MDVRP are well-studied problems, there is a strong intuition to decompose an LRP into its two subproblems, an FLP and an MDVRP. This intuition has been picked up in many LRP heuristics as described in the introduction (Prins et al., 2007; Escobar et al., 2013, 2014; Tuzun and Burke, 1999).

Instead of computing only one FLP solution, one could also compute a number of promising options. More concretely, an LRP could be solved by (1) the determination of promising location options $O_1, O_2, \ldots$, and (2) the computation of delivery routes for each option. This decomposition is visualized in Figure 1. Since the quality of a certain location option can only be determined after a routing solution has been computed, one could also say that the routing solution constitutes an *evaluation* of a location option. If we can execute this evaluation in a short time, we could evaluate and compare a large number of location options.

In the last decades, much research effort has been spent to tackle this problem, and efficiently compute high-quality routing solutions for VRPs and MDVRPs. State-of-the-art routing heuristics can solve MDVRPs with 100 customers almost optimally in a few seconds (Arnold and Sörensen, 2017b; Vidal et al., 2012). In view of this progress, an algorithm that completely decomposes an LRP and purely relies on an effective computation of routing solutions does seem feasible, especially since the evaluation of a location option O_1 does not necessarily require to solve the respective routing problem to optimality. In order to make a statement about whether O_1 represents a good location option (e.g., it is better than O_2), it might be sufficient to solve the corresponding MDVRP up to a certain quality, or in other words, to approximate the routing. The feasibility of this approach relies on the assumption that the number of location options that needs to be evaluated is limited or can be reduced.

2.1. Finding promising location options

The number of candidates for depot locations $F = |C|$ is rather limited in popular benchmark instances. On most instances we have up to 10 possible locations for opening depots, and on some we have up to 20 (Prodhon and Prins, 2014). The resulting number of possible location options seems manageable in comparison to other combinatorial problems. Since we have $\binom{F}{f}$ possibilities to open exactly f depots on F possible locations, the number of location options amounts to $\sum_{f=1}^{F} \binom{F}{f} = 2^F - 1$. For $F = 10$ this adds up to 1023 options. Where usually heuristics have to ignore large parts of the search space, this limited number of options allows us to conduct a complete search through the search space. In fact, we could evaluate all location decisions for $F = 10$ possible locations and $N = 100$ customers with a fast construction heuristic like the one by Clarke and Wright (1964) in a few seconds. However, we could significantly increase the efficiency of a decomposition-based LRP heuristic, and also generalize the heuristic to instances of larger size, if we can exclude unpromising location options before their routing evaluation.

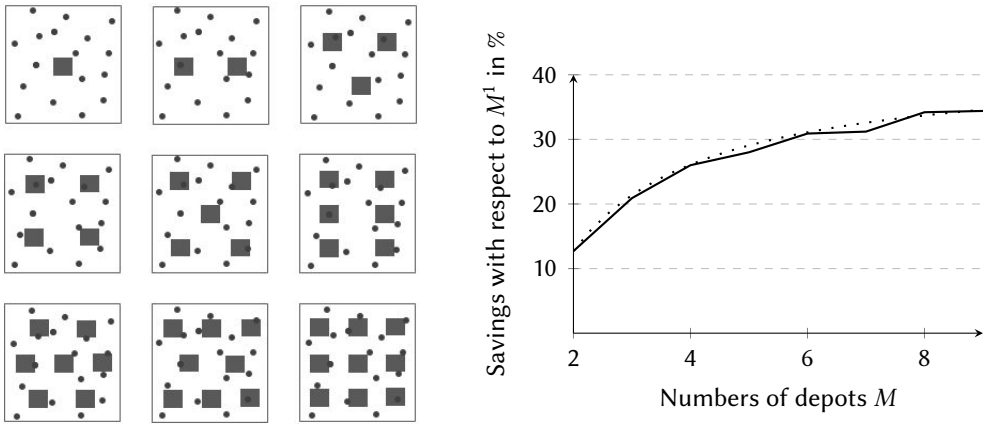

Figure 2: *(right) Average reduction in routing costs when delivering from an increasing number of depots, with respect to delivering from a single depot. The solid line captures the observed data for* $N = 100$*, the dotted line presents the approximated function* $r(M)$*. (left) The depots are distributed uniformly in a grid-pattern.*

Opening depots usually comes at a cost, but it generally can be expected that more open depots enable a more efficient routing. With more open depots, the average distance between customers and depots decreases, assuming the depots are opened at 'good' locations. Therefore, there should be a trade-off between opening costs and routing costs.

We investigate this trade-off in a set of experiments in which we generate and solve MDVRP instances. The instances are generated in such a way that 100 customers are randomly placed in a square and visited from an increasing number of depots on delivery routes that can visit 10 customers on average. To capture the idea that depots are opened in good positions, we locate the depots evenly. Since the customers are located according to a uniform distribution, an even distribution of the depots should minimize the average distance between a customer and its nearest depot. Thus, this setup should represent a decent location option, provided the customers are more or less uniformly distributed in the square. To obtain an even distribution we use a grid pattern, as visualized in Figure 2. The algorithm of how to generate this grid pattern is illustrated in the appendix. Starting with a single central depot, we generate 100 instances and compute solutions with the routing heuristic outlined below, allowing 10 seconds of computation time, and obtain the average routing costs R^1 per instance. We repeat this step with an increasing number of depots M, and compute the reduction in routing costs with respect to a single depot $r(M) = \frac{R^1 - R^M}{R^1}$.

The results in Figure 2 confirm the hypothesis that more open depots result in lower routing costs. Even though the data does not result in a smooth curve, the marginal benefit of another open depot generally decreases with more depots. After experimenting with different models, we found that the observed reduction in routing costs $r(M)$ for M depots with respect to a single open depot can be approximated by the function $r(M) = 1/2^{\frac{1}{M}} - 0.58$. This function is derived on the basis of empirical observations on a specific instance setup and thus presents an approximation, rather than an analytically-proven relationship. For instance, when we repeated the experiments for larger instances with 1,000 customers, we observed that more open depots have greater benefits.

Consequently, the approximation of $r(M)$ is certainly not precise enough to compute the optimal number of open depots for any LRP instance. However, it can be beneficial in the estimation of some upper bound for the number of open depots. For simplicity, assume that each potential depot has the same opening costs c_o, in the case that the opening costs are not constant let c_o be the average opening costs. Then the sum of opening costs increases linearly in the number of open depots, while the marginal reduction in routing costs diminishes with more open depots, as shown in Figure 2. Thus, for some number of open depots, the costs of opening another depot will outweigh the benefits in terms of routing. We can derive an estimate for this upper bound M^U by determining the smallest M for which we have

$$R^1 \cdot (r(M) - r(M-1)) < c_o, \tag{1}$$

where R^1 is an approximation of the routing costs from a central depot. This equation holds, if the estimated savings in terms of routing are lower than the costs of opening a depot, and it is thus not worthwhile to open the depot.

Given a particular LRP instance, we compute M^U by determining the most central depot and by using the simple construction heuristic by Clarke and Wright to estimate R^1. The most central depot is defined as the depot with the smallest average distance to all customers. We can then derive M^U with the above formula and exclude all location options with more than M^U open depots from the search. This cut-off can drastically reduce the number of location options which need to be evaluated. For some benchmark instances with $F = 20$ we obtain $M^U = 3$ (e.g., for instance 112222). Therefore, instead of looking at all possible $2^{20} - 1 = 1,048,575$ location options, we only consider all $\sum_{f=1}^{3} \binom{20}{f} = 6,195$ options with less than 4 depots, a reduction by 99.4%.

If the capacities in the depots are constrained, M^U might constitute too low an

upper bound to obtain a feasible solution. This occurs on the instances by Prins et al. (2006), in which opening costs are relatively high and depots have limited capacities, so that a minimum number of depots have to be opened to supply all customers. In this case, we simply choose the upper bound as the maximum of M^U and the minimal number of open depots with which we can satisfy all demand. If the capacity constraints are relatively loose, the corresponding upper bound is likely to be M^U, but if they are tight, this approach ensures that we do not open more depots than we have to. In Table 3 and in Table 4 in the appendix we list the computed upper bounds for the 36 instances by Tuzun and Burke (1999) and the 30 instances by Prins et al. (2006), together with the number of open depots that we observe in the best solutions in the experiments in Section 4. On most instances without capacitated depots, we observe that the derived upper bounds slightly overestimate the number of open depots in the best solutions, while on most instances with constrained capacities in the depots, as few depots as the constraints allow are opened. We cannot validate that the derived upper bounds hold for all instances, since we do not always obtain the best known solution, and for some instances the optimal solutions are not yet known. However, we obtain high-quality solutions with these upper bounds so that they are likely to include the best location options, while drastically reducing the search space. The remaining location options need to be evaluated with a routing heuristic.

2.2. Efficient approximation of routing solutions

We evaluate promising location options with an effective heuristic based on local search (Arnold and Sörensen, 2017b). Local search tries to iteratively improve a solution by applying small (local) modifications which are called *moves*. In VRPs and MDVRPs, local search moves can either improve a single route (intra-route optimisation) or trigger changes in multiple routes simultaneously (inter-route optimisation). Starting with a solution constructed with the algorithm by Clarke and Wright (1964) (CW), the heuristic uses three three complementary and well-implemented local search operators to improve this initial solution. Individual routes are optimised with the heuristic by Lin and Kernighan (1973) (LK), the optimization of a pair of routes is carried out with the CROSS-exchange operator (CE) (Taillard et al., 1997), and for the simultaneous improvement of more than two routes we propose a relocation chain (RC) which is based on the idea of an ejection chain (Glover, 1996). If no more improvements can be found, the heuristic uses the concept of *guided local search* (GLS) (Voudouris and Tsang, 2003) to escape this local minimum. Rather than changing the respec-

tive solution directly, GLS changes the evaluation of the solution by penalizing edges. The penalization of an edge increases its cost value, and the subsequent local search attempts to remove it. Care should be taken to remove such edges that appear to worsen the solution, and we developed a function $b(\cdot)$ to detect such edges. In summary, the heuristic works in the following way.

1. Construction. Allocate each customer to the nearest depot and construct a starting solution with the heuristic by Clarke and Wright (1964), for each depot separately. Optimise the individual routes with LK.

2. Initial Optimisation. Apply CE and RC on starting solution. Whenever a route is changed, re-optimise it with LK.

Repeat until a certain time limit is reached

3. Perturbation. Repeat until 30 moves have been made.

   3 a. Penalize edge (i, j) according to $b(i, j)$.
   3 b. Try to remove (i, j) with CE and RC.

4. Optimisation. Apply LK, and then iteratively CE and RC on all routes that were changed during perturbation. Whenever a route is changed with CE or RC, re-optimise it with LK.

The heuristic has been shown to be effective for many VRP and MDVRP benchmark instances, and computes near-optimal solutions for MDVRP instances with up to 300 customers in a few seconds. Even though the quality of the computed solutions (also called accuracy) is an important metric, in the context of LRPs we have to evaluate many location options as fast as possible and as accurate as necessary. The evaluation should provide a good indication of whether a location option is better or worse than other options, rather than being perfectly accurate. Since longer computation times typically allow to compute routing solutions of higher quality, the question arising is 'how long do we need to evaluate a location option?'.

To obtain an intuition about the performance behaviour of the heuristic over time, we generated and solved 100 randomly-sampled MDVRP instances with 100 and 200 customers being delivered from two depots. The results in Figure 3 present the average performance of the heuristic over time, with respect to the final solutions computed after 1 minute of computation time. We observe that the CW algorithm computes reasonable solutions with a gap of around 7% almost

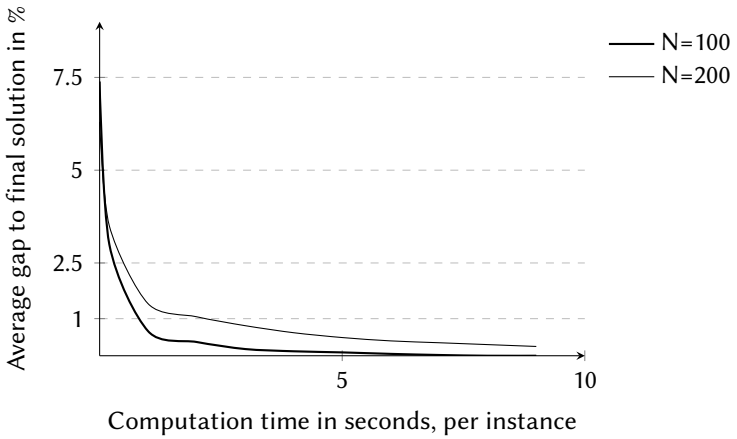

Figure 3: *Average performance of the routing heuristic on 100 random MDVRP instances over time, depending on instance size.*

instantaneously (0.002 seconds for $N = 100$ and 0.004 seconds for $N = 200$). Within the first second of computation time, the heuristic improves the quality of the initial solution significantly, and after two seconds the solutions are usually within a 1% range of high-quality solutions. Hereby, the computational effort to obtain marginal improvements increases with better solutions. In general, for MDVRPs with more customers it requires more time to obtain the same accuracy.

Even though the performance curves might be different for other instance types where customers are, for example, more clustered, these results provide a good intuition about how fast the heuristic is able to obtain a certain solution quality, and thus, how fast and how accurate it can evaluate a certain location option.

3. Solving LRPs through iterative filtering

We have outlined how to use problem knowledge to reduce the number of promising location options in LRPs to a manageable size, and how a state-of-the-art routing heuristic can evaluate these options quite accurately within a few seconds. Together, these findings lead to a simple heuristic design to solve LRPs: evaluate promising location options with a routing heuristic, and choose the best location option, together with the respective routing solution, as the solution for the LRP. However, an accurate evaluation of all promising location options might be time-consuming. As an example, if we allow 10 seconds to compute routes for

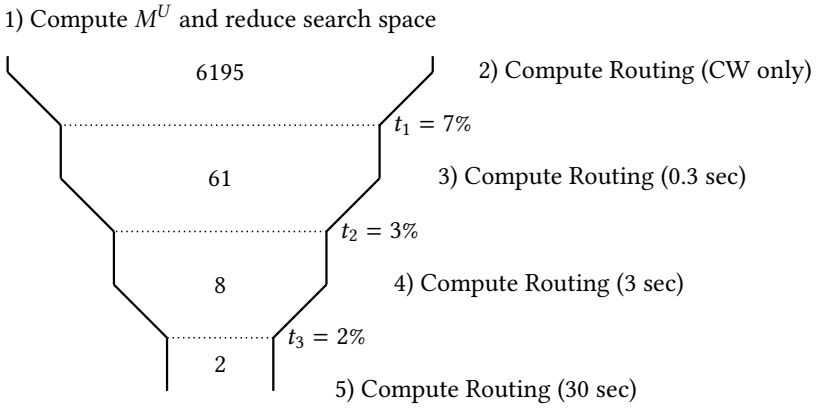

Figure 4: *Illustration of the iterative filtering. At each stage, the routing heuristic is given more time, while less location options are accepted. The number of remaining location options as well as the computation times are taken from instance '111122'.*

each of 6,195 location options in the example above, the heuristic would require more than 17 hours of computation time. On the other hand, an evaluation with CW can be executed in a fraction of a second, but it might be too imprecise to identify the best location option. The time-quality trade-off observed in Figure 3 suggests that an iterative approach might be an efficient compromise. We evaluate a location option until we are sufficiently confident that there exists a better option.

More formally, given a set of location options, we approximate the routing solution for each option. This allows us to approximate c_{LRP} for each option, as well as to determine the best option with costs c^*_{LRP}. We then remove all location options that are not within a $t\%$ range of the best one, i.e., $\frac{c_{LRP}}{c^*_{LRP}} > 1 + t$. The set of location options is, in a manner of speaking, filtered. Less options remain and we can evaluate them more accurately. Iteratively, we reduce the number of remaining location options and increase the accuracy of the routing evaluation. We continue this iterative filtering until only few solutions remain and we can run the routing heuristic for a maximum runtime. More concretely, given an instance with N customers, the iterative filtering involves the following steps.

1) Determine the most central depot and approximate R^1 with CW. Compute M^U with equation 1, obtain an upper bound, and remove all location options in which the number of open depots exceeds the upper bound.

2) For each remaining location option, compute a routing solution with CW

(about 0.002 sec for $N = 100$). Remove all location options with an objective value greater than $t_1 = 7\%$ of the best one (or keep the 100 best).

3) For each remaining location option, compute a routing solution, running the heuristic for $3\frac{N}{1000}$ sec. Remove all location options with an objective value greater than $t_2 = 3\%$ of the best one (or keep the 10 best).

4) For each remaining location option, compute a routing solution, running the heuristic for $3\frac{N}{100}$ sec. Remove all location options with an objective value greater than $t_3 = 2\%$ of the best one (or keep the 3 best).

5) For each remaining location option, compute a routing solution, running the heuristic for $3\frac{N}{10}$ sec. The location option (together with the routing solution) with the best objective value constitutes the final solution for the LRP.

In total, we implement three filtering stages; the first stage uses CW, the second stage allows a runtime of $3\frac{N}{1000}$ seconds, and the final filtering stage allows a runtime of $3\frac{N}{100}$ seconds. Even though the routing heuristic behaves deterministically, note that little variations in computation time can change the outcome of the second and third stage. For the experiments in Section 4 we therefore decided to make the behaviour of these two stage deterministic. In the second filtering stage we apply the initial optimisation step of the routing heuristic, and for the third filter we additionally apply 100 iterations of perturbation and optimisation. The time required for these steps approximately corresponds to the time limits defined for the respective stage.

An overview of this filtering framework is illustrated in Figure 4. This heuristic design is simple, scalable and readily extendible to other problem variants, since the routing heuristic can be treated as a black box. The performance of this framework hinges on the conditions that (1) no stage filters out the best (or one of the best) location option, and that (2) the last stage computes a high-quality routing solution for the remaining options. If both conditions are met, it is likely that the best location option is identified and an near-optional routing solution is computed. In practice, however, it is possible that the best location options only reveal themselves after a certain amount of routing effort and might not pass one of the filters. We therefore also experimented with different thresholds and time limits per filter, however, we observed only minor changes in solution quality. The above setup appears to work well for many instances, and we keep the same parameter configuration for all instance types.

3.1. Capacitated LRPs

One of the most popular variants of LRPs imposes a limit on the demand that can be fulfilled from each depot. More formally, the aggregated demand that is fulfilled on tours originating from a depot must not exceed the capacity limit of the depot. Solving instances of this kind with standard heuristics for MDVRPs could result in infeasible solutions. In Arnold and Sörensen (2017a) we developed an extension for MDVRPs that considers inventory constraints in the depots, and can also distinguish between different products. This problem variant can be reduced to solve MDVRPs with depot capacities, if we consider a single product and the available inventory in the depots corresponds to the capacity of the depots.

The routing heuristic above can be readily adapted to solve instances of this type. The initial solution is created in a greedy fashion, where each customer is delivered by the nearest depot that still has sufficient capacity, and the subsequent local search only considers moves that maintain feasibility with respect to all constraints. Location options in which too few depots are open to deliver all customers can be excluded from the search.

3.2. Large LRPs

The iterative filtering framework relies on the two assumptions that the number of initial location options is manageable, and that the corresponding routing problems can be solved efficiently. In LRPs in which the number of potential depot locations F or the number of customers N is exceedingly large (or both), these assumptions might no longer hold, and the framework might no longer be computationally feasible.

Such very large instances were introduced by Harks et al. (2013), with up to 10,000 customers being delivered from up to 1,000 possible depot locations. To overcome this complexity, we need to reduce the number of initial location options even more drastically, and limit the routing evaluation. Drawing on the intuition from Section 2, if customers are distributed more or less uniformly in the considered plane, then the open depots should be located uniformly as well. Two open depots in close proximity are likely to be less beneficial, than two open depots that are spaced more moderately (given that they have similar opening costs), and in extreme cases all open depots are spaced equidistantly. In other words, we assume that in good location options the open depots are located in a grid-pattern as outlined in Figure 2. The respective algorithm is given in the appendix.

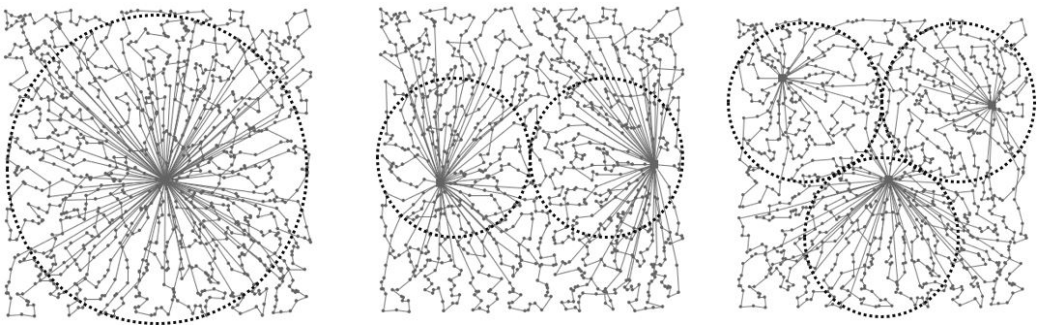

Figure 5: *Location options for large scale LRPs are chosen in a grid-like pattern. For increasing M, we place M circular regions in the plane, and in each region we open one depot, before computing a routing solution with CW. The routing of the best location option generated in this way is improved further.*

We identify such location options, by firstly distributing M points on the plane such that they form a grid. Each point is the center of a circular region, the radius of which equals half the distance to the neighboring point. In each region we try to open one depot. The depot should be close to the center of the region and have low opening costs. Thus, for each depot in the region we compute the sum of the distance to the center and the opening costs, and open the depot with the lowest value. For larger M, the regions become smaller, and it can occur that there is no potential depot in a region. In this case, we simply ignore the region and open less depots in total. After all regions have been investigated, we have generated a location option.

This location option is then evaluated with the heuristic of Clarke and Wright (1964). The time complexity of CW grows quadratically in the number of customers, and thus, for large $N \geq 1000$ we might observe long run-times. In Arnold et al. (2017) we proposed a simple idea to linearize this complexity for very large VRPs, by only considering edges between a customer and its 100 nearest customers. We observed that this approach yields solutions of similar quality while significantly reducing computation time. In summary, for each $M \in \{1, 2, \ldots, 200\}$ we generate a location option, and evaluate this location option with the fast version of CW. This process is illustrated in Figure 5. For simplicity, we choose an upper bound for M of 200 for each instance. We identify the best location option and apply the routing heuristic for $\frac{N}{100}$ seconds. Since a routing evaluation requires more computation time for very large N, we decide not to use any more filters.

This design requires only a small modification of the heuristic above, and, even though it appears simplistic, it has the potential to obtain good LRP solutions in a very short time. It can be extended to obtain better solution, e.g., by taking more location options into consideration or by allowing more computation time for the improvement of the routes.

4. Computational Experiments

We test the performance of the heuristic on the basis of the most popular benchmark sets by Tuzun and Burke (1999) ($\mathbb{T}$) and Prins et al. (2006) ($\mathbb{P}$). The $\mathbb{T}$-instances have between 100 and 200 customers located on a squared plane that have to be delivered from up to 20 potential depot locations with uniform opening costs. The $\mathbb{P}$-instances constitute a complementary benchmark set with capacitated depots and varying opening costs, while the instances are of similar size with 20-200 customers and up to 10 potential depot locations. Additionally, we test the scaling of the heuristic on the recently introduced benchmark set by Harks et al. (2013). These instances comprise between 1,000 and 10,000 customers that can be delivered from up to 1000 potential locations with varying opening costs.

The performance is compared to the most effective LRP heuristics in literature: the GRASP+ILP metaheuristic by Contardo et al. (2014), the granular variable neighborhood search (GVTNS) by Escobar et al. (2014), and the tree-based search algorithm (TBSA) by Schneider and Löffler (2017). These heuristics have only been applied to the first two benchmark sets. The large instances have so far only been tackled by Harks et al. (2013) (Approx+TSP) and Guemri et al. (2016) (2-SH), and we compare with both methods. The above heuristic has been implemented in Java and all tests have been performed within the Eclipse development environment on an AMD Ryzen 3 1300X CPU working at 3.5GHz on Windows 10, using a single thread. To allow a fair comparison of computation times, we normalise the CPU times with respect to the processor (PassMark Software, 2018), as in Schneider and Drexl (2017) (for the comparison on the larger instance set the computation time is less important and we do not normalize here).

The average performance of the above heuristic (denoted A&S) on the LRP benchmark sets are presented in Table 3 and Table 4. For the $\mathbb{T}$-instances and $\mathbb{P}$-instances we present the average gaps with respect to the best known solutions (BKS) as defined in Schneider and Drexl (2017), and for the $\mathbb{H}$-instances the gaps are expressed with respect to the results of this heuristic. All times are given in seconds. The detailed results per instance can be found in the appendix.

Table 1: *Results on the $\mathbb{T}$-instances and the $\mathbb{P}$-instances, based on Schneider and Drexl (2017).*

	GRASP+ILP		GVTNS		TBSA_{speed}		$\text{TBSA}_{quality}$		A&S	
	Gap	Time*	Gap	Time*	Gap	Time*	Gap	Time*	Gap	Time*
$\mathbb{T}$	0.66	1379	0.86	68	0.57	48	0.15	725	0.30	141
$\mathbb{P}$	0.38	619	0.43	31	0.20	23	0.02	452	0.13	55

* normalized according to PassMark Software (2018).

Table 2: *Results on the large $\mathbb{H}$-instances*

Approx+TSP		2-SH		A&S (only CW)		A&S (improved routes)	
Gap	Time	Gap	Time	Gap	Time	Gap	Time
12.42	221	11.01	66	5.37	93	0.00	146

GRASP+ILP and TBSA include stochastic components and, thus, we compare with the average performance of these heuristics, whereas all other heuristics are deterministic and obtain the same solution after every run.

For the smaller benchmark sets, we observe that the filtering heuristic computes high-quality solutions that are very close to the best known solutions for almost all instances. Overall, the heuristic achieves a better accuracy than GVTNS and GRASP+ILP and a similar performance than TBSA on both benchmark sets. All of these solutions are computed in short computation times, which place the heuristic among the most efficient ones in literature. The corresponding quality–time trade-off is highlighted in Figure 6.

The heuristic can also tackle large problems successfully, as demonstrated by the results on the $\mathbb{H}$-instances. It improves the best known results on almost all instances by more than 10% on average in a comparable computation time. Notably, the relatively simple approach of opening depots in a grid-like pattern and constructing routes with CW already improves the previous solutions by more than 5%.

In summary, not only is the heuristic the first which can be successfully applied to LRPs of different problem sizes, but it also constitutes one of the most efficient heuristics for most LRP benchmark instances in literature. At the same time, its simple design allows for a scalable and flexible implementation. The heuristic can also be readily parallelized, since the evaluation of location options

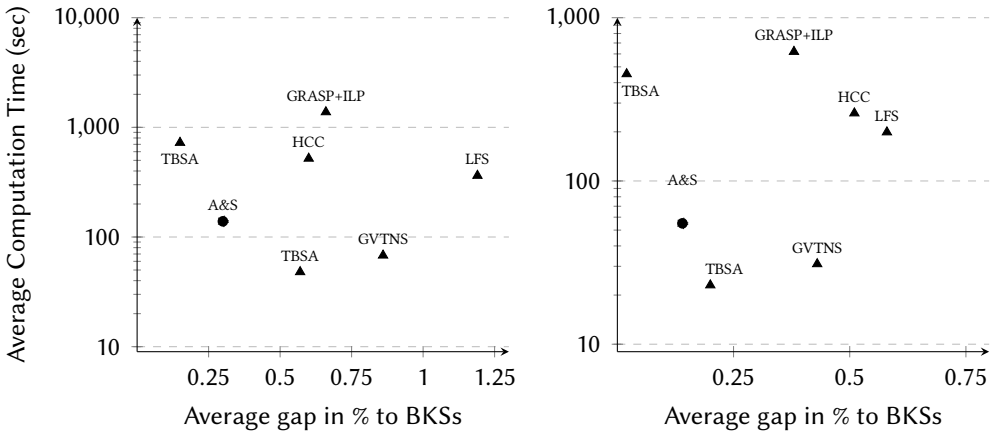

Figure 6: *Performance versus computation time in comparison to state-of-the-art heuristics on the $\mathbb{T}$-instances (left) and the $\mathbb{P}$-instances (right), based on Schneider and Drexl (2017).*

at each filtering stage can be executed simultaneously. This parallelization would increase efficiency further.

5. Conclusion

In this paper we have designed a heuristic for LRPs that uses a routing heuristic to iteratively filter promising location options. We estimate an upper bound for the number of open depots to reduce the number of location options, which are then evaluated with a routing heuristic. The improvements found by the routing heuristic decline over time, and thus, we use a filtering framework to iteratively decrease the number of promising options in different stages. In each stage, the accuracy of the computed routing solutions increases. The same heuristic, with some minor modifications, can also be used to solve LRPs with capacitated depots and large LRPs. Computational tests on benchmark sets show that the resulting heuristic can effectively solve a wide range of LRP instances within short computation times.

The heuristic design is flexible and scalable, and therefore useful in practical cases where implementation time is limited. A straightforward CW implementation is sufficient to obtain satisfactory results, while a state-of-the art routing heuristic is able to compete with the best results in literature. In more general terms, these findings suggest that the reduction of an LRP to a routing problem constitutes a successful heuristic approach. It decomposes the problem and results in a straightforward heuristic design. The same approach could be used to

solve other LRP variants, e.g., multi-echolon LRPs, or other problems that revolve around routing, e.g., the multi-trip VRP.

References

Arnold, F., Gendreau, M., and Sörensen, K. (2017). Efficiently solving very large scale routing problems. Working paper 75, Polytechnique Montreal, CIRRELT.

Arnold, F. and Sörensen, K. (2017a). From storage to shipment - the effect of ignoring inventory when planning routes. Working paper 2, University of Antwerp, Faculty of Applied Economics.

Arnold, F. and Sörensen, K. (2017b). A simple, deterministic, and effcient knowledge-driven heuristic for the vehicle routing problem. Working paper 12, University of Antwerp, Faculty of Applied Economics.

Clarke, G. and Wright, J. W. (1964). Scheduling of vehicles from a central depot to a number of delivery points. *Operations research*, 12(4):568–581.

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

Cordeau, J.-F., Gendreau, M., Laporte, G., Potvin, J.-Y., and Semet, F. (2002). A guide to vehicle routing heuristics. *Journal of the Operational Research society*, 53(5):512–522.

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

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

Glover, F. (1996). Ejection chains, reference structures and alternating path methods for traveling salesman problems. *Discrete Applied Mathematics*, 65(1):223–253.

Guemri, O., Beldjilali, B., Bekrar, A., and Belalem, G. (2016). Two-stage heuristic algorithm for the large-scale capacitated location routing problem. *International Journal of Mathematical Modelling and Numerical Optimisation*, 7(1):97–119.

Harks, T., König, F. G., and Matuschke, J. (2013). Approximation algorithms for capacitated location routing. *Transportation Science*, 47(1):3–22.

Lin, S. and Kernighan, B. W. (1973). An effective heuristic algorithm for the traveling-salesman problem. *Operations research*, 21(2):498–516.

PassMark Software (2018). Cpu benchmarks. `https://www.cpubenchmark.net/`. Accessed: 2018-02-05.

Prins, C., Prodhon, C., and Calvo, R. W. (2006). 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.

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

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

Salhi, S. and Rand, G. K. (1989). The effect of ignoring routes when locating depots. *European journal of operational research*, 39(2):150–156.

Schneider, M. and Drexl, M. (2017). A survey of the standard location-routing problem. *Annals of Operations Research*, 259(1-2):389–414.

Schneider, M. and Löffler, M. (2017). Large composite neighborhoods for the capacitated location-routing problem. *Transportation Science*.

Taillard, É., Badeau, P., Gendreau, M., Guertin, F., and Potvin, J.-Y. (1997). A tabu search heuristic for the vehicle routing problem with soft time windows. *Transportation science*, 31(2):170–186.

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

Vidal, T., Crainic, T. G., Gendreau, M., Lahrichi, N., and Rei, W. (2012). A hybrid genetic algorithm for multidepot and periodic vehicle routing problems. *Operations Research*, 60(3):611–624.

Voudouris, C. and Tsang, E. P. (2003). *Guided local search*. Springer.

Algorithm 1 Placing M points in a square with length L

1: $cols \leftarrow \lceil \sqrt{M} \rceil$
2: $rows \leftarrow \lceil \frac{M}{cols} \rceil$
3: $shortRows \leftarrow cols \cdot rows - M$
4: $d_x \leftarrow \frac{L}{cols}$
5: $d_y \leftarrow \frac{L}{rows}$
6: $makeShort \leftarrow true$
7: **if** $shortRows < \frac{rows}{2}$ **then**
8: $\quad makeShort \leftarrow false$
9: **end if**
10: $cur_y \leftarrow \frac{d_y}{2}$
11: $cur_{row} \leftarrow 0$
12: **while** $cur_{row} < rows$ **do**
13: $\quad$ **if** $shortRows > 0$ AND $makeShort$ **then**
14: $\quad\quad points \leftarrow cols - 1$
15: $\quad\quad cur_x \leftarrow d_x$
16: $\quad\quad makeShort \leftarrow false$
17: $\quad\quad shortRows \leftarrow shortRows - 1$
18: $\quad$ **else**
19: $\quad\quad points \leftarrow cols$
20: $\quad\quad cur_x \leftarrow \frac{d_x}{2}$
21: $\quad\quad makeShort \leftarrow true$
22: $\quad$ **end if**
23: $\quad cur_{col} \leftarrow 0$
24: $\quad$ **while** $cur_{col} < points$ **do**
25: $\quad\quad placePoint(cur_x, cur_y)$
26: $\quad\quad cur_x \leftarrow cur_x + d_x$
27: $\quad\quad cur_{col} \leftarrow cur_{col} + 1$
28: $\quad$ **end while**
29: $\quad cur_y \leftarrow cur_y + d_y$
30: $\quad cur_{row} \leftarrow cur_{row} + 1$
31: **end while**

Table 3: *Results on the $\mathbb{T}$-instances by Tuzun and Burke (1999). Gap in % to the BKS, time in seconds, estimated upper bounds of open depots U and obtained number of open depots M.*

Instance	BKS	GRASP+ILP			GVTNS			A&S				
		Value	Gap	Time	Value	Gap	Time	Value	Gap	Time	M	U
111112	1467.68	1475.50	0.53	198	1479.21	0.79	84	1468.29	0.04	101	3	4
111122	1448,37	1452.00	0.25	580	1485.28	2.55	126	1449.20	0.06	104	2	4
111212	1394.80	1405.80	0.79	220	1402.59	0.56	74	1394.80	0.00	105	2	4
111222	1432.29	1440.60	0.58	755	1463.23	2.16	99	1432.29	0.00	104	2	4
112112	1167.16	1176.20	0.77	278	1167.16	0.00	83	1167.16	0.00	35	2	4
112122	1102.24	1103.60	0.12	634	1102.24	0.00	105	1102.24	0.00	67	2	4
112212	791.66	795.80	0.52	227	791.66	0.00	96	791.66	0.00	35	2	3
112222	728.30	728.50	0.03	550	728.30	0.00	126	728.30	0.00	66	2	3
113112	1238.44	1239.60	0.11	286	1238.49	0.02	82	1238.49	0.02	97	3	4
113122	1245.30	1246.30	0.09	646	1247.27	0.17	127	1245.31	0.00	104	3	4
113212	902.26	902.80	0.06	231	902.26	0.00	71	902.26	0.00	65	3	4
113222	1018.29	1018.29	0.00	749	1018.29	0.00	85	1018.29	0.00	104	3	4
131112	1892.17	1924.10	1.68	1640	1933.67	2.19	179	1892.17	0.00	149	3	5
131122	1819.68	1831.00	0.62	3612	1852.14	1.78	173	1826.69	0.39	162	4	4
131212	1960.02	1969.30	0.47	1275	1983.09	1.18	184	1960.02	0.00	110	3	4
131222	1792.77	1800.30	0.43	3099	1803.01	0.58	175	1792.77	0.00	161	3	5
132112	1443.32	1450.40	0.49	871	1443.43	0.20	186	1446.53	0.22	146	2	4
132122	1429.30	1447.20	1.25	2738	1441.43	0.84	210	1444.66	1.07	118	2	4
132212	1204.42	1205.90	0.12	2082	1204.42	0.00	128	1204.85	0.04	145	3	4
132222	924.68	931.90	0.78	3734	931.28	0.71	177	931.43	0.73	52	3	4
133112	1694.18	1703.80	0.57	938	1701.34	0.42	182	1709.26	0.89	148	3	4
133122	1392.01	1401.50	0.68	2751	1416.74	1.78	175	1400.50	0.61	170	3	4
133212	1197.95	1199.60	0.13	1010	1213.87	1.32	207	1200.24	0.19	96	3	4
133222	1151.37	1158.70	0.64	3560	1151.80	0.04	208	1156.61	0.46	159	3	4
121112	2237.73	2251.30	0.61	2805	2258.02	0.91	315	2255.26	0.78	205	3	5
121122	2137.45	2154.90	0.82	5680	2166.20	1.35	300	2142.73	0.25	286	4	5
121212	2195.17	2226.10	1.41	3004	2239.65	2.03	287	2201.94	0.31	202	4	5
121222	2214.86	2241.70	1.21	6143	2236.73	0.99	351	2218.88	0.18	281	4	5
122112	2070.43	2093.80	1.13	3462	2103.82	1.61	278	2071.42	0.05	214	3	5
122122	1685.52	1704.40	1.12	8547	1717.92	1.92	433	1694.66	0.54	326	3	5
122212	1449.93	1467.80	1.23	3471	1469.45	1.35	318	1449.92	0.00	73	2	4
122222	1082.46	1086.70	0.39	5292	1082.46	0.00	349	1082.87	0.04	171	3	5
123112	1942.23	1986.70	2.29	3865	1969.38	1.40	261	1965.70	1.21	155	4	5
123122	1910.08	1936.20	1.37	9367	1935.74	1.34	344	1915.30	0.27	288	4	5
123212	1761.11	1766.20	0.29	3766	1776.90	0.90	349	1800.44	2.23	132	3	4
123222	1390.86	1392.70	0.13	5157	1391.50	0.05	317	1391.71	0.06	152	5	5

Table 4: *Results on the $\mathbb{P}$-instances by Prins et al. (2006). Gap in % to the BKS, time in seconds, estimated upper bounds of open depots U and obtained number of open depots M.*

Instance	BKS	GRASP+ILP			GVTNS			A&S				
		Value	Gap	Time	Value	Gap	Time	Value	Gap	Time	M	U
20-5-1a	54793	54793	0.00	2	54793	0.00	2	54793	0.00	6	3	3
20-5-1b	39104	39104	0.00	3	39104	0.00	3	39104	0.00	6	2	2
20-5-2a	48908	48908	0.00	1	48945	0.08	2	48908	0.00	6	3	3
20-5-2b	37542	37542	0.00	3	37542	0.00	3	37542	0.00	6	2	2
50-5-1a	90111	90111	0.00	15	90111	0.00	13	90111	0.00	17	3	3
50-5-1b	63242	63281	0.06	18	63242	0.00	9	63242	0.00	18	2	3
50-5-2a	88293	88333	0.05	18	89342	1.19	12	88298	0.01	32	3	3
50-5-2b	67308	67436	0.19	22	67951	0.96	10	67308	0.00	47	3	3
50-5-2bis	84055	84055	0.00	21	84126	0.08	8	84055	0.00	33	3	4
50-5-2bbis	51822	51898	0.15	27	52213	0.75	9	51822	0.00	32	3	3
50-5-3a	86203	86203	0.00	17	86203	0.00	18	86203	0.00	17	2	3
50-5-3b	61830	61853	0.04	23	61885	0.09	20	61830	0.00	48	2	3
100-5-1a	274814	275628	0.30	220	276137	0.48	75	275450	0.23	35	3	3
100-5-1b	213568	214785	0.57	230	216154	1.21	59	213967	0.19	35	3	3
100-5-2a	193671	194054	0.20	122	193896	0.12	76	193671	0.00	32	2	2
100-5-2b	157095	157311	0.14	100	157180	0.05	82	157150	0.04	32	2	2
100-5-3a	200079	200394	0.16	97	200777	0.35	69	200127	0.02	32	2	2
100-5-3b	152441	152814	0.24	100	153435	0.65	68	152441	0.00	33	2	2
100-10-1a	287661	292657	1.74	2622	287864	0.07	203	289449	0.62	112	3	3
100-10-1b	230989	236026	2.18	1067	232599	0.70	117	230989	0.00	68	3	3
100-10-2a	243590	243851	0.11	236	245484	0.78	52	243590	0.00	64	3	3
100-10-2b	203988	204253	0.13	259	204252	0.13	42	203988	0.00	65	3	3
100-10-3a	250882	253610	1.09	723	254558	1.47	82	252890	0.80	102	3	3
100-10-3b	203114	205110	0.98	584	205824	1.33	78	204567	0.72	37	3	3
200-10-1a	474850	477656	0.59	3960	477009	0.45	320	475225	0.08	132	3	3
200-10-1b	375177	378656	0.93	4006	377716	0.68	239	376884	0.45	195	3	3
200-10-2a	448077	449797	0.38	4943	449006	0.21	231	449124	0.23	64	3	3
200-10-2b	373696	374996	0.35	3486	374717	0.27	290	374192	0.13	68	3	3
200-10-3a	469433	471272	0.39	4075	471978	0.54	330	471183	0.37	132	3	3
200-10-3b	362320	363581	0.35	7888	362827	0.14	214	362748	0.12	133	3	3

Table 5: *Results on the large scale instances by Harks et al. (2013). Gap in % to the computed results, time in seconds and obtained number of open depots M.*

Instance	Approx+TSP			2-SH			A&S			
	Value	Gap	Time	Value	Gap	Time	Value	Gap	Time	M
M 1,1	13,478.9	22.13	1	12,014.2	8.87	1	11,035.6	0.00	17	50
M 1,2	3,499.1	10.30	1	3,470.3	9.38	1	3,172.3	0.00	16	16
M 1,3	2,478.9	1.28	2	2,814.5	15.01	11	2,446.8	0.00	16	4
M 2,1	17,997	20.01	1	15,760.5	5.10	1	14,995.7	0.00	15	33
M 2,2	4,468.7	9.08	1	4,494.1	9.71	1	4,096.2	0.00	15	6
M 2,3	2,620.8	1.78	2	3,010.6	16.93	11	2,754.1	0.00	15	1
M 3,1	22,926.8	23.49	1	19,839.5	6.86	1	18,565.6	0.00	15	18
M 3,2	5,345.9	11.05	1	5,207.4	8.18	1	4,813.2	0.00	15	6
M 3,3	2,779.3	3.64	1	3,156.6	17.70	11	2,681.4	0.00	15	1
L 1,1	32,325.9	15.89	36	29,538.1	5.90	65	27,893.1	0.00	98	177
L 1,2	8,106.1	6.50	59	8,176.0	7.42	20	7,611.0	0.00	96	52
L 1,3	5,463.7	0.84	122	6,228.4	14.96	67	5,417.7	0.00	98	14
L 2,1	50,229.7	24.00	41	43,496.5	24.00	64	40,508.1	0.00	103	89
L 2,2	12,059.5	13.13	74	11,551.6	8.36	19	10,659.6	0.00	104	19
L 2,3	6,624.8	10.60	165	7,058.2	17.85	69	5,989.2	0.00	100	4
L 3,1	63,905.1	26.55	51	54,976.6	8.87	64	50,495.9	0.00	133	55
L 3,2	14,372.4	15.63	98	13,426.2	8.02	20	12,429.4	0.00	100	15
L 3,3	6,966.5	8.60	213	7,467.2	16.41	66	6,414.4	0.00	117	4
XL 1,1	48,677.1	7.99	214	43,939.8	-2.52	179	45,074.3	0.00	265	193
XL 1,2	11,872	8.19	369	11,867.3	8.14	89	10,973.3	0.00	225	95
XL 1,3	7,754.7	1.21	750	8,824.0	15.16	161	7,661,6	0.00	218	25
XL 2,1	77,580.6	22.51	243	68,118.7	7.57	171	63,325.5	0.00	324	168
XL 2,2	18,159.1	10.76	435	17,638.8	7.58	89	16,395,0	0.00	234	25
XL 2,3	9,296.1	6.42	887	10,172.32	16.45	163	8,734.9	0.00	354	5
XL 3,1	101,454	29.03	307	85,509.5	8.75	178	78,628.1	0.00	372	108
XL 3,2	21,341.5	12.57	570	20,659.7	8.97	89	18,958.5	0.00	280	18
XL 3,3	10,389.9	12.21	1323	10,897.5	17.68	167	9,259.5	0.00	390	5