ERASMUS UNIVERSITY ROTTERDAM

ERASMUS SCHOOL OF ECONOMICS

BACHELOR THESIS ECONOMETRICS AND OPERATIONS RESEARCH

Knowledge-Guided Local Search For The Vehicle Routing Problem

Author
Daniël Hak
Student Number
481709

Supervisor
D. GALINDO PECIN
Second Assessor
P. C. BOUMAN

Abstract

Local search is a well known way to solve vehicle routing problems. In this paper I will propose a well working heuristic that implements local search methods and get to quality solutions in a relative short time. The heuristic does not make use of a random component. The local search methods are combined with a perturbation to get good solutions. This paper also shows how the heuristic can be adjusted and be used for variations of the vehicle routing problem.

July 5, 2020

Contents

1 Introduction

The vehicle routing problem (VRP) is one of the most studied combinatorial optimization problems. Given a set of customers, the objective is to minimize the traveling cost, while visiting all the customers. All vehicles start and end at a depot and visit customers to deliver their demand. For companies it is interesting to minimize their transportation cost, because most of the times this cost is a significant amount of the total cost for a product. Using optimized routes, companies can save on the transportation cost. An efficient heuristic to solve the VRP is proposed by Arnold and Sörensen (2019a) to obtain high quality solutions. They want to solve the capacitated vehicle routing problem, where each vehicle can deliver a limited amount of goods, all vehicles have the same capacity. They propose the knowledge-guided local search method (KGLS) which combines three local search methods in an efficient way. In combination with perturbation this leads to solid solutions. The VRP is a NP-hard problem, so using effective solution methods is necessary to obtain high quality solutions. For delivery companies the VRP is a relevant well known problem. For these companies an efficient solution method is key to address this challenging combinatorial problem. The method proposed in this paper does not make use of random components, other type of heuristics have some random components used in their heuristics. It is interesting to see how the KGLS performs compared to these heuristics. Other variants of the vehicle routing problem will be addressed and solved with the KGLS, these variants are the vehicle routing problem with time windows (VRPTW) and the vehicle routing problem with a heterogeneous fleet.

In order to find good solutions for the vehicle routing problem the following research question is formulated:

- Does the knowledge-guided local search method also provide good solutions for other variants of the vehicle routing problem?

The research question is supported by the following sub-questions:

- How does the local search method perform compared to other state-of-the-art solution methods?
- Does the knowledge-guided local search method give a good starting point for a local search method for a vehicle routing problem with time windows and for a vehicle routing problem with a heterogeneous fleet?

- Can the knowledge-guided local search method simply be adjusted for other vehicle routing variants?

The research question together with the sub-questions form the core of this paper. This paper first shows some effective local search methods and how they are combined in a algorithm for a VRP. It also shows how it can be applied to a vehicle routing problem with time windows (VRPTW) and on a vehicle routing problem with heterogeneous fleet.

The paper is organised as follows. In section 2 state-of-the-art solution methods are given for the VRP. Also know solution methods for the VRPTW and VRP with heterogeneous fleet are shown. Section 3 gives a mathematical formulation of the VRP. In section 4 the KGLS heuristic is given and explained, in section 5 the usage of KGLS is extended to other variants. In section 6 the used data is explained. In section 7 the obtained results are given and analysed. Section 8 gives a brief summary of the findings.

2 Literature

In Johnson, Papadimitriou, and Yannakakis (1988) the effectiveness of local search is explained, the local optimums are easier to find then optimal solutions of the problem. VRP can be optimized by inter- or intra-route optimizations. A well known intra-route optimization method is given in Lin and Kernighan (1973) (LK), where κ-opt moves efficiently used to optimize a single route. The LK heuristic will be further explained in 4.2. In Subramanian et al. (2012) a iterated local search (ILS) heuristic is proposed, the heuristic makes use of a random selected neighbourhood. ILS is combined with a Set Partitioning approach to improve the solution even more. In Toth and Tramontani (2008) a solution method is proposed where a initial solution is optimized by first selecting a neighbourhood and reducing the size of the neighbourhood with heuristics. Then the reduced neighbourhood is optimized by using the ILP formulation with Column Generation problem. The neighbourhood is chosen with random components using probability functions based on several aspects like the distance of the edges.

As an addition to the normal vehicle routing problem, time windows can be implemented. In Bräysy and Gendreau (2005) multiple local search methods are compared, the heuristics in Bräysy (2002) and Russell (1995) are recommended as good and efficient solution methods in this paper.

In Bräysy (2002) a three phase approach is suggested. In stage one an initial solution is made, in the second stage the number of routes is be minimized. The third stage uses an Or-opt exchange proposed by Or (1977). In Russell (1995) a hybrid algorithm between a interchange heuristic and a parallel construction heuristic is suggested. The interchange heuristic is embedded in the parallel construction heuristic to get quality solutions.

The fleet in a VRP could be heterogeneous, the cost and capacity of the vehicles could differ between the types. In Subramanian et al. (2012) a hybrid algorithm is proposed where a iterated local search based heuristic is combined with set partitioning formulation. Random neighbourhood ordering is used and Or-opt moves are used for intra-route optimization. For unlimited fleets an empty route of each vehicle type is added. In Gendreau et al. (1999) a tabu search heuristic is proposed with a random chosen neighbourhood. When a vertex is moved from a route to another route, it is not allowed to be reinserted for the next θ iterations, where θ is randomly selected from an interval. After the tabu search a optimization and fleet change is executed on the best known solution. In Taillard et al. (1997) initially the problems are solved as homogeneous VRP's, this solutions are then effectively combined to solve the problem with a heterogeneous fleet by choosing routes that are often generated in the homogeneous solutions. The routes are selected in a way that they have no customers in common. Then the solution is optimized with a tabu search method and afterwards this solution is further solved in CPLEX.

3 VRP formulation

The VRP problem can be notated in a mathematical formulation. First the variables and parameters of the capacitated VRP are introduced and explained. Then the objective function is given with the accessory restrictions. After the capacitated VRP formulation the formulation of the respectively the VRPTW and VRP with heterogeneous fleet are given.

3.1 CVRP

The vehicle routing problem can be seen as a complete graph $G = (V, E)$, with V the set of customers and a depot, E is the set of edges between the customers. The variable x indicates if the edge between customer i and j is used by vehicle k. Parameter c gives the cost for travelling between customer i and j. Set K is introduced as the set of available vehicles, all vehicles are

homogeneous.

$$x_{i,j,k} = \begin{cases} 1 & \text{if edge } (i,j) \in E \text{ is used by vehicle } k. \\ 0 & \text{otherwise.} \end{cases}$$

$c_{i,j} =$ The cost to travel from customer i to customer j.

$d_i =$ The demand from customer i.

$q =$ The capacity of a vehicle.

With these variable and parameters the following objective function and constraints are formulated. The first node in V is the depot node.

$$\text{min.} \quad \sum_{k \in K} \sum_{i \in V} \sum_{j \in V} c_{i,j} x_{i,j,k} \tag{1}$$

$$\text{s.t.} \quad \sum_{k \in K} \sum_{i \in V} x_{i,j,k} = 1 \qquad \forall j \in V \setminus \{0\} \tag{2}$$

$$\sum_{i \in V} x_{i,h,k} = \sum_{j \in V} x_{h,j,k} \qquad \forall h \in C \setminus \{0\}, \forall k \in K \tag{3}$$

$$\sum_{i \in V} x_{i,0,k} = \sum_{j \in V} x_{0,j,k} \qquad \forall k \in K \tag{4}$$

$$\sum_{i \in V} d_i \sum_{j \in V} x_{i,j,k} \leq q \qquad \forall k \in K \tag{5}$$

$$x_{i,j,k} \in \mathbb{B} \qquad \forall i,j \in V, \forall k \in K \tag{6}$$

The objective function (1) minimizes the total travelling cost, most of the times the cost between customer i and j is the euclidean distance between the nodes. Constraint (2) states that every customer is visited exactly once. Constraints (3) and (4) ensure that every vehicle that leaves the depot also returns and that after arriving at a customer the vehicle also leaves the customer. Constraint (4) also ensures that the amount of vehicles leaving the depot is equal to the amount of vehicles entering the depot. Finally the constraint (5) makes sure that every route does not exceed the vehicle capacity.

3.2 CVRPTW

For CVRPTW the customers now have a limited time interval when they have to be served. The objective function is the same as for CVRP, but extra restrictions have to be added. First a new variable and parameters are introduced. Variable s indicates when vehicle k arrives at customers i. Parameters a and b indicate the time window and e is the service time. The travelling time between customer i and j is here equal to the distance.

$$
\begin{aligned}
s_{i,k} &= \text{ The time when vehicle } k \text{ arrives at customer } i. \\
a_i &= \text{ The opening time of customer } i. \\
b_i &= \text{ The closing time of customer } i. \\
e_i &= \text{ The service time of customer } i. \\
M &= \text{ Large number that exceeds the time horizon.}
\end{aligned}
$$

The following restrictions are added to the normal CVRP model.

$$
s_{i,k} + c_{i,j} + e_i - M(1 - x_{i,j,k}) \leq s_{j,k} \qquad \forall i, j \in V, \forall k \in K \tag{7}
$$

$$
a_i \leq s_{i,k} \leq b_i \qquad \forall i \in V, \forall k \in K \tag{8}
$$

$$
s_{i,k} \in \mathbb{B} \qquad \forall i \in V, \forall k \in K \tag{9}
$$

Here constraint (7) states that the time that vehicle k cannot arrive at customer j before the arrival time at customer i plus the travel and service time if vehicle k travels from customer i to j. Constraint (8) ensures that the time windows are respected.

3.3 VRP with heterogeneous fleet

For VRP with heterogeneous fleet the different types of vehicles have different costs, so the objective function is slightly different. All the normal restrictions of the CVRP have to be respected. The set of vehicles K now has a specific demand and cost for each k. One variable and two parameters are introduced for this formulation. Variable y indicates whether vehicle k is used. Parameter q indicates now the capacity and f indicates the cost for vehicle k.

$$y_k = \begin{cases} 1 & \text{if vehicle } k \in K \text{ is used.} \\ 0 & \text{otherwise.} \end{cases}$$

$q_k =$ The capacity of vehicle k.

$f_k =$ The cost of vehicle k.

Now the objective function is different and constraint (5) is slightly adjusted. One constraint is added to the model. All the other restrictions of CVRP are the same.

$$\text{min.} \quad \sum_{k \in K} (f_k y_k + \sum_{i \in V} \sum_{j \in V} c_{i,j} x_{i,j,k}) \tag{10}$$

$$\text{s.t.} \quad \sum_{i \in V} d_i \sum_{j \in V} x_{i,j,k} \leq q \qquad \forall k \in K \tag{11}$$

$$\sum_{j \in V} x_{0,j,k} = y_k \qquad \forall k \in K \tag{12}$$

$$y_k \in \mathbb{B} \qquad \forall k \in K \tag{13}$$

The objective function (10) now minimizes the travelling cost combined with the vehicle cost. Constraint (11) is the adjusted version of (5), where now for every vehicle type a different capacity is used. Constraint (12) ensures that a vehicle only leaves the depot when it is used.

4 Arnold and Sörensen heuristic

The knowledge-guided local search algorithm is separated into a construction, an initial optimisation, a perturbation and an optimisation part. In the construction part the Clarke and Wright (1964) (CW) heuristic is used. After CW the individual routes are optimized with the heuristic from Lin and Kernighan (1973) (LK). The initial optimisation will consist of Cross-Exchange (CE) and Relocation Chain (RC). The routes changed are re-optimised with LK and the "badness" of the edges is set, the badness of an edge is explained later on. Then the perturbation starts and the edge with the highest badness is penalized. CE and RC are applied with the new costs of the edges. After 30 edges are penalized, LK is applied and then CE and RC with the normal costs. If a route is changed after CE or RC, it will be re-optimized with LK and the badness will be adapted. The perturbation and re-optimization process are continued until the time limit is reached. In this section first the local search methods used in this heuristic are explained. Then the implementa-

tion and construction of the algorithm is given. A pseudo-code of the algorithm can be found in the appendix at subsection 9.4.

4.1 Local search

Local search methods are a well known solution method for NP-hard problems. For a VRP multiple local search operators can be used. There are two types of operators: inter-route operators and intra-route operators. In inter-route optimization the allocation of costumers between routes is optimized, in intra-route optimization the route itself is optimized. A well known way of intra-route optimization is 2-opt. In 2-opt two edges are removed and replaced by two new edges, the two edges are chosen so that the total cost is lower. In 2-opt the replacement results in a reversed order for a part of the tour. The sequence A-B-C-D-E-F-G-H-A for example changes to A-B-C-D-G-F-E-H-A if the edges between (D,E) and (H,G) are removed and replaced with (D,G) and (E,H) if this results in a better solution. The optimization is better described in Lin and Kernighan (1973). For inter-route optimization the swap, Or-exchange and a crossover are three well known methods to change the allocation of customers between routes. In a swap four edges are exchanged in such a way that a subset of customers from route one are allocated to route two and a subset of customers from route two is allocated to route one. In an Or-exchange a subset of customers from route one is allocated to route two, by changing three edges. In a crossover two edges are exchanged and a subset of customer from route one is allocated to route two, a subset of customers from route two is allocated to route one in return, both subsets are connected to the depot. The inter-route optimization will be explained in more detail in section 4.3.

4.2 Lin-Kernighan heuristic

Intra-route optimization is the same as a traveling salesman problem. A well known intra-route optimization method is the Lin-Kernighan heuristic (LK). LK uses κ-opt moves to improve the route. With κ-opt moves the complexity increases fast, so for this problem we use a upper limit for κ of four. With κ-opt moves edges are removed and added to come to a better solution. With LK we start looking at the longest edge in the route, this edge is considered to be removed. If we want to remove the edge another edge that shares an endpoint with the removed edge has to be added. When looking for a edge to add, we look at edges to the ten nearest nodes. We only consider a move if the partial gain criterion is satisfied, so the cost of the added edges has to be lower than

the cost of the removed edges. The pair of edges also has to satisfy the condition that closing the tour results in a feasible tour. In a VRP the routes contains most of the times a few customers, so all feasible κ-opt moves are considered for a particular starting edge before a move is executed. The set of removed and added edges has to be distinct and for each node in the route only one of the initial edges may be removed. When for a starting edge an improvement is found the move with the highest gain is executed and LK will start again until no improvements can be found. An example of a 3-opt move is given in figure 3. A more complete description can be found in Lin and Kernighan (1973) or in Helsgaun (2000).

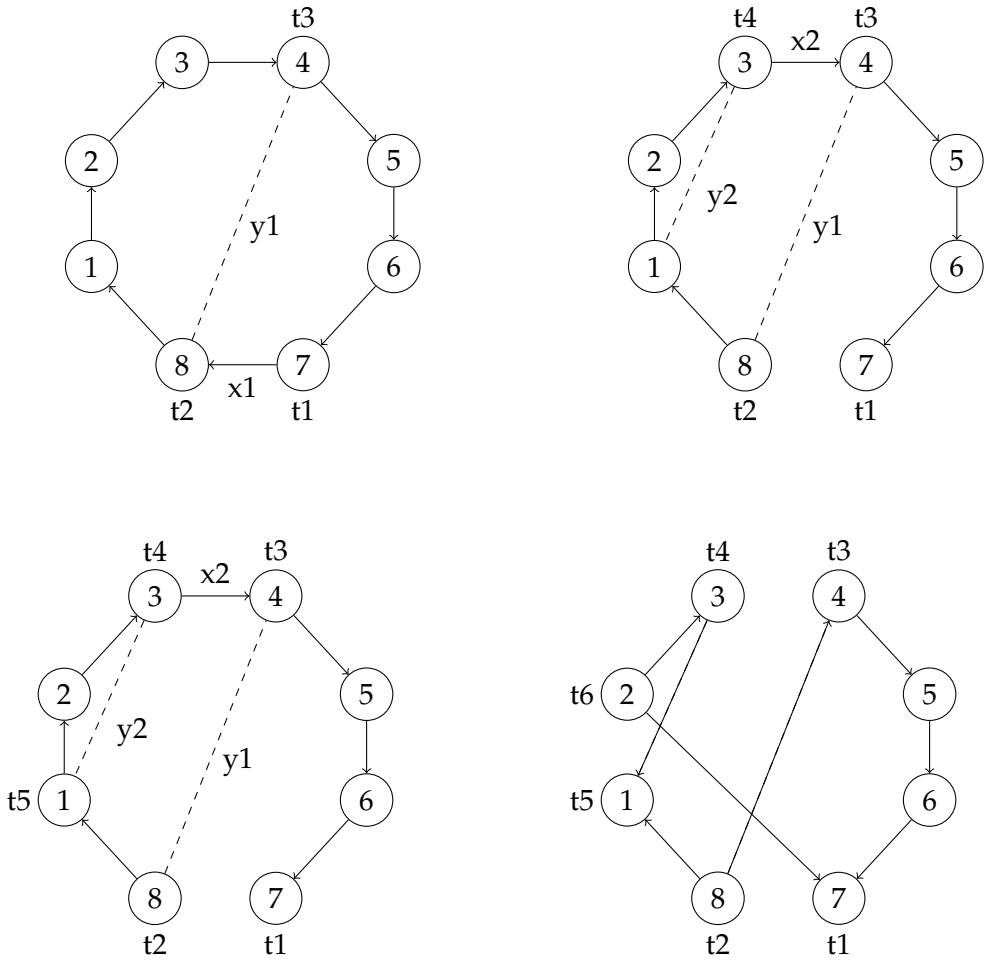

Figure 1: Example of a LK move with a 3-opt move. First the edge (7,8) is removed and the edge (8,4) is added. This must result in a positive gain to consider continuation of the move. An edge incident to 4 must be removed and adding an edge from the new t4, edge (3,4) is removed and edge (1,3) is added. Again continuation is only considered if the gain is positive. The move can be completed by removing edge (1,2) and adding edge (2,7). The move is executed if it is the highest possible gain.

4.3 CROSS-exchange

CROSS-exchange (CE) is a inter-route optimizer, where two substrings $\hat{r_i}$ and $\hat{r_j}$ of routes r_i and r_j are exchanged if this leads to a positive gain. For CE we first need to determine node I_k as the starting point from the substring $\hat{r_i}$, I_k is the node in route I on index k of that route. For I_k we only consider the C closest nodes as the starting point J_l of $\hat{r_j}$. When we consider a particular node J_l, we need to remove edge to an incident node of J_l, either (J_l, J_{l-1}) or (J_l, J_{l+1}) and the other end of the removed edge is connected to I_{k+1}. This is visually shown in figure 4.3. We only consider the substring if the start results in a positive gain. So for the removal of edge (I_k, I_{k+1}), we need to consider 4C starts, 2C for both nodes. Whenever the starting move results in a positive gain other lengths of the substring are considered. If the completion of the exchange results in a positive gain the move is added to the possible moves. Every size of substring $\hat{r_j}$ is considered until the depot is reached or the capacity constraint is violated. For every size of $\hat{r_i}$ the same consideration applies. There are also special cases where either $\hat{r_i}$ or $\hat{r_j}$ is empty, here different starting moves apply. If for example $\hat{r_i}$ is empty we only consider the cost of removed edge (I_k, I_{k+1}) and added edge (I_k, J_l). If all possible substring are considered the candidate with the steepest decent is executed. A more elaborate description of CE is given in Taillard et al. (1997).

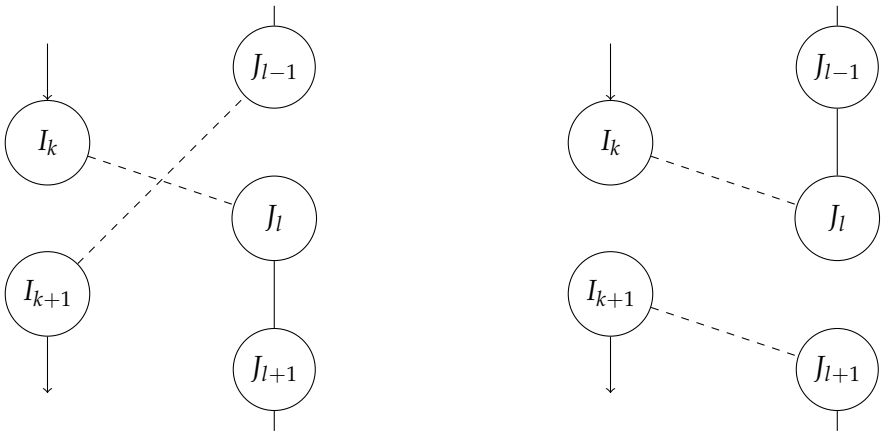

Figure 2: Example of the possible starts for a given I_k and I_{k+1}, substring $\hat{r_i}$ start in the direction of J_{l+1} or J_{l-1}.

4.4 Relocation chain

Relocation chain (RC) is a also a inter-route optimizer. RC starts with the relocation of one customer from route r_i to route r_j. A node from route r_j is then replaced from r_j to route r_k. We repeat this process until we have reached the limit of relocations. If the moves result in a positive gain, the move is executed. If the move from a node from r_i to r_j results in a violation of the capacity constraint, the move can still become feasible if a node from r_j is replaced into route r_k. The cost of the detour of node I_k from route r_i can be computed by:

$$c_D = c(I_{k-1}, I_{k+1}) - c(I_k, I_{k-1}) - c(I_k, I_{k+1})$$

Customer I_k will be inserted next to node J_l, it could be before or after J_l. The option with the least cost will be considered. The cost of the insertion can therefore be computed with:

$$c_1 = min_{J_l^* \in (J_{l-1}, J_{l+1})}(c(I_k, J_l) + c(I_k, J_l^*) - c(J_l, J_l^*)$$

If $C_D + C_1 \leq 0$ it can be considered as a start. If the relocation results in a feasible move the move is considered as a candidate, else a relocation for a customer in r_j is considered. The candidate move with the highest gain will be executed. The limit of relocations is set at three.

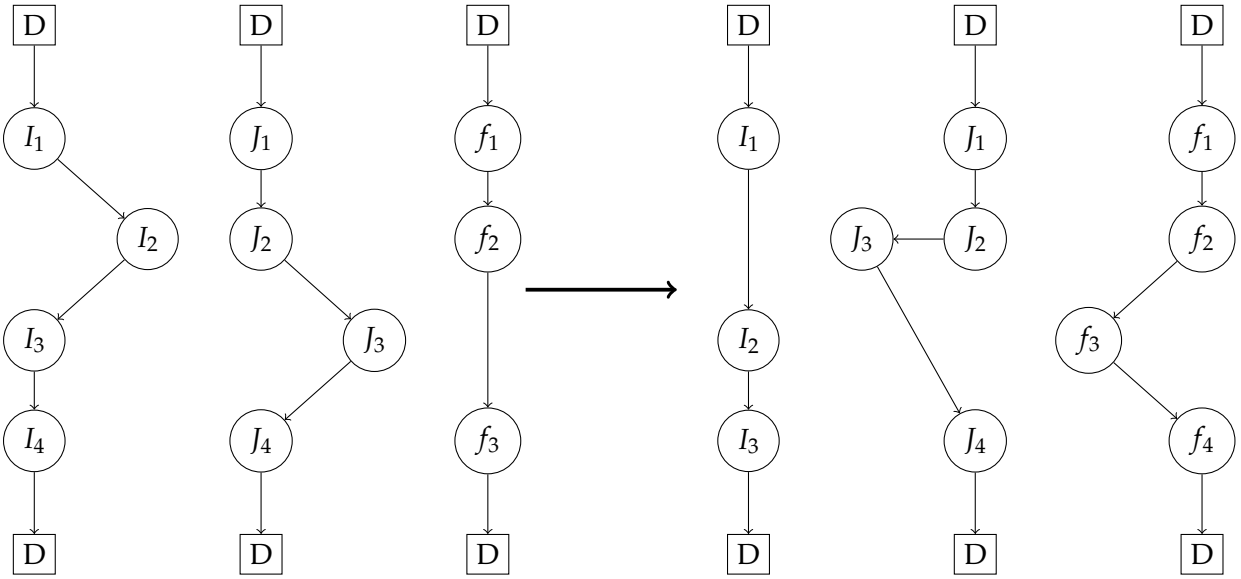

Figure 3: Example of a RC move with 2 relocations. First the customer I_2 relocated to route J after J_2 , then customer J_3 is relocated to route f after customer f_2.

4.5 Knowledge-guided local search heuristic

4.5.1 Initial solution

The construction begins with finding a initial solution with CW. The good solutions mostly have as few routes as possible, because for every extra route we have an extra edge. To make sure the starting solution begins with as few routes as possible we set a lower bound of different routes at $M_{min} = \lceil \frac{D}{Q} \rceil$, where D is the total demand and Q is the capacity limit. CW computes a solution with M_{CW} routes, if $M_{CW} > M_{min} + 1$ then we will compute a new CW solution with a modification. The classic CW is computed with the savings:

$$s(i,j) = c(d,i) + c(d,j) - c(i,j)$$

Here d is the depot and c(i,j) is the cost of the edge between customer i and customer j. With the modified CW the savings will be replaced with weighted savings. The weighted saving for edge (i,j) is computed as follows:

$$s_w(i,j)\frac{s(i,j)}{max_{k,l}s(k,l)} + \frac{d(i)+d(j)}{max_{k,l}(d(k)+d(l))}$$

Here s(i,j) is computed as stated above and d(i) denotes the demand of customer i. Customers with a high demand are now prioritized with this new formula, this should lead to less vehicles needed to carry the demand. The individual routes in the starting solution are optimized with LK.

After the construction, initial optimisation is performed. CE and RC are applied to the solution, after these two are applied all routes that changed are re-optimized with LK. The initial optimisation is now finished and badness of an edge can be set. To compute the badness of an edge we use the following formula:

$$b^w(i,j) = \frac{w(i,j)}{1+p(i,j)}$$

Here w(i,j) is the width of an edge and p(i,j) the number of received penalties, which will be initialized at 0 for every edge. The width of an edge is computed as explained in Arnold and Sörensen (2019b). Now everything is set up and the perturbation can start.

4.5.2 Perturbation

When the perturbation starts the edge with the highest badness is penalized with one. Then we use CE and RC on the edge (i,j) and we use c^g as evaluation criterion. Here c^g for edge (i,j) is computed as follows:

$$c^g(i,j) = c(i,j) + \lambda p(i,j) L$$

Here c(i,j) is the normal cost of edge (i,j), lambda decides the impact of penalization and experiments showed that lambda = 0.01 was a good value. L represents the average cost of a edge in the starting solution. After this process is repeated 30 times the current solution is optimized. First LK is applied, then CE and RC are applied on all edges that were changed during perturbation. Now we use the normal cost as criterion. If a route is changed with CE or RC we apply LK on the route. Now we update all the b^w values. This process is repeated until the time limit is reached.

5 Extensions

The KGLS can be extended to solve other variants of the VRP. I demonstrate how the algorithm could be adjusted so good quality solutions can be provided for the VRP with time windows and the VRP with a heterogeneous fleet.

5.1 Time windows

The knowledge-guided local search algorithm can also be used to tackle the vehicle routing problem with time windows. With time windows, a customer can only be visited in a specific time frame. This adds another constraint to the problem, which must be satisfied. Now the heuristics for the initial solution have to be performed with another feasibility check. The time window constraints have to be respected as well.

For the following steps in the knowledge-guided local search, CE and RC can still be used, but they have to take the time windows into account as well now. For every possible move now a feasibility check for the time windows have do be done before a move is executed. The LK heuristic can results in a big change in the route order, so when the last move is considered the new possible solution is tested in both orders on feasibility for time windows. If the move is feasible and also results in a positive gain the move is considered as a candidate. For CE a move results in allocating

customers from route one into route two and vice versa, a possible move is checked on feasibility before it is considered as a candidate. For RC all relocations for customers to another route are also checked on feasibility for time windows. Because of the time windows, some parameter values could now no longer be optimal and could be considered to be optimized again with experimental research. The optimal size of the neighbourhood or the penalization value could vary now.

5.2 Heterogeneous fleet

An other variant and a extension of the vehicle routing problem is the vehicle routing problem with with heterogeneous fleet. In the problem multiple sorts of vehicles can be used. Now the algorithm must have the ability to change the vehicle type of a route. First with the initial solution CW is adjusted to decide which type of vehicles are in the starting solution. The savings are now computed with:

$$s(i,j)^* = c(d,i) + c(d,j) - c(i,j) + C(r_i) + C(r_j) - C(r_i + r_j)$$

The first three terms correspond with the normal CW savings. The term $C(r_i)$ gives the cost of the smallest possible vehicle to carry the demand of the route whereto i belongs. After every move in CW the last three terms can differ, because adding a node to that specific route, so the saving for every customer are computed again until no more moves can be executed. This solution should offer a good starting point for the KGLS. Now KGLS should also be able to change the vehicle type when optimizing, to achieve this, after the normal RC is executed a adjusted version of RC will start. This version is able to relocate a customer from one route to another while changing the vehicle type for both routes if necessary. In this relocation only one customer is relocated. The relocating is executed if this results in a positive gain, taking the costs of the vehicles into account as well. This version of RC will also be executed in the perturbation part.

Because KGLS does penalize the cost of a vehicle type, the penalty cost for edges should not be to high, because then the cost of the vehicles will not be significant enough and the algorithm will tend to use only the largest type of vehicle. Experiments showed that using $\lambda = 0.001$ gave the best results.

6 Data

The data is obtained from Uchoa et al. (2017). The data contains 100 instances varying from 100 till 1000 customers. The instances differ also on other characteristics, there is variation in the customer distribution, the vehicle capacity, the demand of the customers and the relative position of the depot. The position of the customers can be randomly chosen or be clustered, some instances have a combination of the two. The capacity varies so different average lengths of routes are present in the solutions. This data set provides a wide variation of possibilities so every type of problem could be investigated. The results will be compared with the results from Arnold and Sörensen (2019a).

For the VRPTW, the 56 instances with 100 customers from Solomon (1987) are used. The 100 customers are uniformly or in a cluster distributed over a square. The instances are divided into short and long time windows. In the instances with short time windows, the capacity of a vehicle will be small, so only a few customers can be visited. The solutions computed in Kallehauge et al. (2005) will be set as the benchmark for the knowledge-guided local search algorithm.

For VRP with heterogeneous fleet the benchmark from Taillard et al. (1997) will be used. The instances used are from Christofides and Eilon (1969) and extended with different types of vehicles by Golden et al. (1984). This are instances of 75 to 100 customers with three types of vehicles.

7 Results

The KLGS is implemented in Java using Eclipse and the instances are executed on a Intel Core i7 processor. For the instances a time limit is set, as a time limit per 100 customers KGLS is performed 3 minutes, so the time limit is computed with $3\frac{N}{100}$ minutes, where N is the number of customers. In pre-testing using higher time limits did not result in significantly better solutions. Experiments showed that using $\lambda = 0.01$ gave the best result form CVRP and CVRPTW, using $\lambda = 0.001$ gave the best results for VRP with heterogeneous fleet. A pseudo-code of KGLS can be found in the appendix at subsection 9.4.

7.1 Replication

A summary of the result for the VRP are given in table 1. All individual results can be found in the appendix in table 4, 5, 6 and 7.

Table 1: Average results of instances from Uchoa et al. (2017). Time is reported in seconds and the gap is in respect to the specific benchmark.

		Arnold and Sörensen	
N	Time	Gap	Time
100-250	109	0.81%	104
251-500	312	0.93%	266
501-1001	935	1.23%	596
	446	0.99%	319

The table shows that the results obtained are slightly worse then the results from Arnold and Sörensen (2019a). Also the time for computing the best solutions is slightly higher. For instances with more customers the gap with the solution value becomes larger. A possible explanation for this gap is the penalization of the edges, in the text it is unclear if the edge (i,j) is considered as the same edge as (j,i). In LK or CROSS exchange the order of some used edges can turn, this could lead to slower penalization of some edges. The paper also did not explain in which order CE must be executed, so the difference in solution values may be due to the different order of the CE execution. Now the edges were sorted on their saving value. The computational times were significantly higher then the times of Arnold and Sörensen (2019a). This might be due to a more efficient way of programming. Also not every instance had the same amount of vehicles in the optimal solution.

7.2 Extensions

For VRPTW the instances of Solomon (1986) were tested. In table 2 the results for all the different types are given. In Kallehauge et al. (2005) the solutions for the instances that are optimally solved are given. Not for every instance the optimal solution was known. The solutions for all instance can be found in the appendix in tables 8, 9 and 10.

Table 2: Results of the instances given in Solomon (1986). Gap given is the difference between the solution of KGLS and the solution as given in Kallehauge et al. (2005). Time is given in seconds, the computational time for the results in Kallehauge et al. (2005) where not given.

	Gap	Time KGLS
Clustered	1.32%	18.7
Random	1.94%	105.5
Random-Clustered	2.74%	96.8

The sets of instances had a very different quality of results. The first set of clustered instances had a for all instances a gap smaller then 0.22%. The second set of clustered instances scored significantly worse with some outliers of more then 5%. The sets of random instances scored mostly between 1 and 2 percent with some outliers to 4%. The set of random clustered scored overall the worst with almost all instances having a gap of more than 2%. The gaps are computed to their optimal solution, for some instances the optimal solution was not known in Kallehauge et al. (2005).

For the VRP with heterogeneous fleet edges should be penalized less, because they are only a part of the total cost. In KGLS the vehicles for a specific route are not penalized, this resulted in a lower optimal λ. The results for the instances from Golden et al. (1984) are given in table 3. They are compared to the results from Taillard et al. (1997). The results from Taillard et al. (1997) are given in the best solution and the average solution computed, the time is the average computed time.

Table 3: Results of the instances given in Golden et al. (1984). The first gap given is the difference between the solution of KGLS and the best solution in Taillard et al. (1997). The second gap is the difference between KGLS and the average solution in Taillard et al. (1997). Time is given in seconds.

	Taillard et al			
N	Time	Gap Best	Gap Average	Time
50-100	57	2.52 %	2.19 %	2648

KGLS performs slightly worse for most instances. For instance 14 the gap between the prices of the vehicles are significantly higher than for most other instance, here KGLS performs worse than in most other cases. KGLS computes the solutions much faster then the algorithm given in Taillard et al. (1997). Most instances scored or between 0% and 0.5% or between 2% and 3%. Their were

no differences in solution qualities compared to the size of instance.

8 Conclusion

The aim of this paper was to test if KGLS could provide good solutions for variants of the VRP. KGLS was compared with several state-of-the-art solution methods to know how good KGLS performed. Three local search methods were combined to get good solutions. The solutions were slightly worse than the used benchmarks, but the computation times were significantly lower. The KGLS could thus be used for problems were low computation times are needed.

To know if KGLS was good starting point for VRPTW and VRP with heterogeneous fleet, the solutions of KGLS were compared with high quality benchmarks. The results of KGLS were slightly worse, but also here the computation times were lower. For VRP with heterogeneous fleet the types of vehicles were not included in the penalization. In some cases this made KGLS perform significantly worse.

KGLS also could easily be adjusted to perform well for other variants. The design of KGLS is straight forward so implementations for feasibility checks or new relocation methods can be easily implemented. The VRPTW needed an extra feasibility check for potential relocations. For VRP with heterogeneous fleet an extra option to change the vehicle type was implemented in relocation chain. In most cases this worked well, but for vehicles with a high cost difference this method performed significantly worse.

In the future KGLS could be used as a starting-point for other solving methods and could be improved with smarter penalization methods. For the other VRP with heterogeneous fleet a better way to switch from vehicles could be implemented, or the vehicles should be penalized as well, this could improve the solution quality. For CVRPTW KGLS could be adjusted so it also tries to minimize the waiting time. KGLS is a well working solution method for problems with a restricted computation time.

References

[1] Florian Arnold and Kenneth Sörensen. "Knowledge-guided local search for the vehicle routing problem". In: *Computers & Operations Research* 105 (2019), pp. 32–46.

[2] Florian Arnold and Kenneth Sörensen. "What makes a VRP solution good? The generation of problem-specific knowledge for heuristics". In: *Computers & Operations Research* 106 (2019), pp. 280–288.

[3] Olli Bräysy. "Fast local searches for the vehicle routing problem with time windows". In: *INFOR: Information Systems and Operational Research* 40.4 (2002), pp. 319–330.

[4] Olli Bräysy and Michel Gendreau. "Vehicle routing problem with time windows, Part I: Route construction and local search algorithms". In: *Transportation science* 39.1 (2005), pp. 104–118.

[5] Nicos Christofides and Samuel Eilon. "An algorithm for the vehicle-dispatching problem". In: *Journal of the Operational Research Society* 20.3 (1969), pp. 309–318.

[6] Geoff Clarke and John W Wright. "Scheduling of vehicles from a central depot to a number of delivery points". In: *Operations research* 12.4 (1964), pp. 568–581.

[7] Michel Gendreau, Gilbert Laporte, Christophe Musaraganyi, and Éric D Taillard. "A tabu search heuristic for the heterogeneous fleet vehicle routing problem". In: *Computers & Operations Research* 26.12 (1999), pp. 1153–1173.

[8] Bruce Golden, Arjang Assad, Larry Levy, and Filip Gheysens. "The fleet size and mix vehicle routing problem". In: *Computers & Operations Research* 11.1 (1984), pp. 49–66.

[9] Keld Helsgaun. "An effective implementation of the Lin–Kernighan traveling salesman heuristic". In: *European Journal of Operational Research* 126.1 (2000), pp. 106–130.

[10] David S Johnson, Christos H Papadimitriou, and Mihalis Yannakakis. "How easy is local search?" In: *Journal of computer and system sciences* 37.1 (1988), pp. 79–100.

[11] Brian Kallehauge, Jesper Larsen, Oli BG Madsen, and Marius M Solomon. "Vehicle routing problem with time windows". In: *Column generation*. Springer, 2005, pp. 67–98.

[12] Shen Lin and Brian W Kernighan. "An effective heuristic algorithm for the traveling-salesman problem". In: *Operations research* 21.2 (1973), pp. 498–516.

[13] Ilhan Or. "TRAVELING SALESMAN TYPE COMBINATORIAL PROBLEMS AND THEIR RELATION TO THE LOGISTICS OF REGIONAL BLOOD BANKING." In: (1977).

[14] Robert A Russell. "Hybrid heuristics for the vehicle routing problem with time windows". In: *Transportation science* 29.2 (1995), pp. 156–166.

[15] Marius M Solomon. "Algorithms for the vehicle routing and scheduling problems with time window constraints". In: *Operations research* 35.2 (1987), pp. 254–265.

[16] Marius M Solomon. "On the worst-case performance of some heuristics for the vehicle routing and scheduling problem with time window constraints". In: *Networks* 16.2 (1986), pp. 161–174.

[17] Anand Subramanian, Puca Huachi Vaz Penna, Eduardo Uchoa, and Luiz Satoru Ochi. "A hybrid algorithm for the heterogeneous fleet vehicle routing problem". In: *European Journal of Operational Research* 221.2 (2012), pp. 285–295.

[18] Éric Taillard, Philippe Badeau, Michel Gendreau, François Guertin, and Jean-Yves Potvin. "A tabu search heuristic for the vehicle routing problem with soft time windows". In: *Transportation science* 31.2 (1997), pp. 170–186.

[19] Paolo Toth and Andrea Tramontani. "An integer linear programming local search for capacitated vehicle routing problems". In: *The vehicle routing problem: Latest advances and new challenges*. Springer, 2008, pp. 275–295.

[20] Eduardo Uchoa, Diego Pecin, Artur Pessoa, Marcus Poggi, Thibaut Vidal, and Anand Subramanian. "New benchmark instances for the capacitated vehicle routing problem". In: *European Journal of Operational Research* 257.3 (2017), pp. 845–858.

9 Appendix

9.1 VRP results

Table 4: Results given of instances from Uchoa et al. (2017), instance number is given with number of customers between brackets. Difference is computed between KGLS and the solution in Arnold and Sörensen (2019a). Reported time is given in seconds.

			Arnold and Sörensen		
Instance	KGLS	Time	Solution	Gap	Time
1 (101)	27643.6	38	27650.0	-0.02%	180
2 (106)	26504.3	30	26411.0	0.35%	90
3 (110)	15017.3	41	14971.0	0.31%	6
4 (115)	12764.4	76	12747.0	0.14%	12
5 (120)	13399.2	104	13332.0	0.50%	6
6 (125)	56441.8	11	55798.0	1.15%	6
8 (134)	10999.0	55	10916.0	0.76%	12
9 (139)	13661.7	17	13590.0	0.53%	6
10 (143)	15866.4	252	15728.0	0.88%	18
11 (148)	43934.1	10	43599.0	0.77%	102
12 (153)	21950.1	262	21390.0	2.62%	78
13 (157)	16911.0	3	16876.0	0.21%	252
14 (162)	14287.7	31	14147.0	0.99%	66
15 (167)	20684.8	38	20589.0	0.47%	12
16 (172)	45974.0	50	45807.0	0.36%	60
18 (181)	25743.3	6	25628.0	0.45%	42
19 (186)	24401.7	76	24194.0	0.86%	132
20 (190)	17164.0	195	17036.0	0.75%	324
21 (195)	44634.1	50	44442.0	0.43%	96
22 (200)	59931.8	75	58747.0	2.02%	300
23 (204)	19754.8	177	19666.0	0.45%	216
24 (209)	30919.3	51	30738.0	0.59%	30
25 (214)	11092.4	177	10929.0	1.50%	150
26 (219)	117796.4	50	117696.0	0.09%	66
27 (223)	41041.5	31	40691.0	0.86%	96

Table 5: Results given of instances from Uchoa et al. (2017), instance number is given with number of customers between brackets. Difference is computed between KGLS and the solution in Arnold and Sörensen (2019a). Reported time is given in seconds.

			Arnold and Sörensen		
Instance	KGLS	Time	Solution	Gap	Time
28 (228)	25983.4	356	25830.0	0.59%	108
29 (233)	19517.7	346	19341.0	0.91%	204
30 (237)	27282.6	176	27146.0	0.50%	12
31 (242)	83500.3	99	83144.0	0.43%	378
32 (247)	38439.8	355	37336.0	2.96%	240
33 (251)	39090.4	29	38916.0	0.45%	66
34 (256)	19011.3	28	18899.0	0.59%	126
35 (261)	26919.7	237	26704.0	0.81%	390
36 (266)	76464.7	57	75966.0	0.66%	282
37 (270)	35591.9	87	35453.0	0.39%	12
38 (275)	21487.1	156	21300.0	0.88%	132
39 (280)	34009.0	429	33709.0	0.89%	474
40 (284)	20577.2	500	20389.0	0.92%	258
41 (289)	96075.9	125	95885.0	0.20%	396
42 (294)	47846.7	78	47450.0	0.84%	258
43 (298)	34608.8	373	34332.0	0.81%	18
44 (303)	22089.9	212	21877.0	0.97%	186
45 (308)	26505.8	295	26072.0	1.66%	162
46 (313)	95475.4	187	94844.0	0.67%	534
47 (317)	78840.8	110	78414.0	0.54%	432
48 (322)	30627.6	217	30038.0	1.96%	156
49 (327)	27985.1	583	27652.0	1.20%	240
50 (331)	31572.5	588	31142.0	1.38%	372
51 (336)	140653.9	415	141060.0	-0.29%	24
52 (344)	42851.2	546	42398.0	1.07%	486
53 (351)	26438.1	214	26162.0	1.06%	180
54 (359)	52381.1	471	51988.0	0.76%	24
55 (367)	23721.9	339	22972.0	3.26%	498
56 (376)	148209.0	145	147879.0	0.22%	522
57 (384)	66949.9	174	66400.0	0.83%	42

Table 6: Results given of instances from Uchoa et al. (2017), instance number is given with number of customers between brackets. Difference is computed between KGLS and the solution in Arnold and Sörensen (2019a). Reported time is given in seconds.

			Arnold and Sörensen		
Instance	KGLS	Time	Solution	Gap	Time
58 (393)	38655.9	364	38381.0	0.72%	78
59 (401)	66792.0	276	66571.0	0.33%	30
60 (411)	20271.6	698	20065.0	1.03%	642
61 (420)	109036.7	164	108351.0	0.63%	150
62 (429)	66412.1	430	65820.0	0.90%	240
63 (439)	36709.9	693	36502.0	0.57%	252
64 (449)	56555.0	394	55747.0	1.45%	72
65 (459)	24760.3	275	24240.0	2.15%	810
66 (469)	226375.7	536	223433.0	1.32%	138
67 (480)	90944.7	340	89970.0	1.08%	228
68 (491)	67670.4	463	67261.0	0.61%	654
69 (502)	69621.3	156	69327	0.42%	900
70 (513)	24638.5	700	24287	1.45%	132
71 (524)	158465.8	625	155342	2.01%	606
72 (536)	96180.6	678	95875	0.32%	156
73 (548)	87507.1	396	86920	0.68%	672
74 (561)	43403.3	1008	42989	0.96%	78
75 (573)	51444.2	970	51020	0.83%	630
76 (586)	193396.0	1054	191094	1.20%	210
77 (599)	110261.0	389	109397	0.79%	642
78 (613)	60960.9	298	60100	1.43%	156
79 (627)	63255.4	602	62612	1.03%	132
80 (641)	65256.3	510	64035	1.91%	144
81 (655)	107168.0	318	106969	0.19%	132
82 (670)	152107.8	1205	147796	2.92%	18
83 (685)	69590.2	458	68927	0.96%	288
84 (701)	83479.0	710	82551	1.12%	144
85 (716)	44234.4	745	43772	1.06%	1008
86 (733)	137639.5	830	137364	0.20%	606
87 (749)	78909.7	652	78337	0.73%	156

Table 7: Results given of instances from Uchoa et al. (2017), instance number is given with number of customers between brackets. Difference is computed between KGLS and the solution in Arnold and Sörensen (2019a). Reported time is given in seconds.

			Arnold and Sörensen		
Instance	KGLS	Time	Solution	Gap	Time
88 (766)	117755.4	1372	115418	2.03%	588
89 (783)	73992.2	1008	73038	1.31%	708
90 (801)	74499.4	947	73525	1.33%	912
91 (819)	161333.1	1471	159525	1.13%	888
92 (837)	196497.3	1496	195203	0.66%	834
93 (856)	89909.5	1536	89289	0.69%	1494
94 (876)	100888.4	1493	100244	0.64%	1194
95 (895)	55643.5	1404	54321	2.43%	264
96 (916)	334614.2	1643	331081	1.07%	648
97 (936)	139024.3	1400	133968	3.77%	1578
98(957)	86572.6	1341	85756	0.95%	882
99 (979)	120665.5	1478	119737	0.78%	576
100 (1001)	74814.8	1015	73060	2.40%	1686

9.2 VRPTW

Table 8: Results of instance from Solomon (1986). First value represents the distance, second value the total scheduling time and third the waiting time. The difference with the optimal solutions given in Kallehauge et al. (2005) is computed. Times of solutions given in Kallehauge et al. (2005) were not given.

			Kallehauge et al					Kallehauge et al	
Instance	KGLS	Time	Solution	Gap	Instance	KGLS	Time	Solution	Gap
C101	828.9	1.2	827.3	0.20%	C201	591.6	1.5	589.1	0.42%
	9328.6					9530.7			
	0.0					0.0			
C102	828.9	2.5	827.3	0.20%	C202	591.6	12.5	589.1	0.42%
	9339.5					9526.7			
	0.0					0.0			
C103	828.1	7.1	826.3	0.21%	C203	644.8	40.4	588.7	9.53%
	9423.6					9600.0			
	445.8					592.3			
C104	824.8	81.5	822.9	0.23%	C204	614.5	10.6	588.1	4.48%
	9653.2					9576.3			
	1040.3					781.1			
C105	828.9	2.9	827.3	0.20%	C205	589.7	4.3	586.4	0.57%
	9342.7					9528.8			
	0.0					0.0			
C106	828.9	1.8	827.3	0.20%	C206	590.4	52.5	586	0.74%
	9328.6					9529.4			
	0.0					0.0			
C107	828.9	2.9	827.3	0.20%	C207	590.2	14.7	585.8	0.75%
	9328.6					9601.4			
	0.0					267.4			
C108	828.9	7.3	827.3	0.20%	C208	607.2	34.5	585.8	3.66%
	9328.6					9544.2			
	0.0					0.0			
C109	828.9	40.7	827.3	0.20%					
	9336.7								
	0.0								

Table 9: Results of instance from Solomon (1986). First value represents the distance, second value the total scheduling time and third the waiting time. The difference with the optimal solutions given in Kallehauge et al. (2005) is computed. Times of solutions given in Kallehauge et al. (2005) were not given.

			Kallehauge et al					Kallehauge et al	
Instance	KGLS	Time	Solution	Gap	Instance	KGLS	Time	Solution	Gap
R101	1646.2	145.6	1637.7	0.52%	R201	1156.8	31.9	1143.2	1.19%
	2545.2					6111.6			
	2420.0					5764.5			
R102	1489.2	14.8	1466.6	1.54%	R202	1055.2	91.0		
	2247.6					5564.8			
	1929.8					4993.1			
R103	1231.3	49.5	1208.7	1.87%	R203	879.4	128.0		
	2159.3					4262.2			
	1403.2					3455.4			
R104	1007.8	142.5	971.5	3.74%	R204	750.3	115.0		
	1901.6					3452.7			
	506.2					2179.6			
R105	1382.2	172.0	1355.3	1.98%	R205	971.1	128.6		
	1955.4					3296.2			
	1052.7					971.1			
R106	1262.2	167.3	1234.6	2.24%	R206	900.7	78.5		
	1922.4					3438.4			
	430.2					2335.2			
R107	1079.3	178.6	1064.6	1.38%	R207	818.3	146.7		
	1886.2					2673.3			
	391.6					1416.9			
R108	958.1	63.7			R208	710.9	83.6		
	1921.5					2136.6			
	548.5					696.0			
R109	1170.2	125.1	1146.9	2.03%	R209	877.3	49.8		
	1935.6					2908.1			
	536.5					1608.5			
R110	1111.8	49.0	1068	4.10%	R210	924.3	95.5		
	1961.4					4551.9			
	610.9					3765.8			
R111	1057.0	146.9	1048.7	0.79%	R211	771.5	159.0		
	1903.4					2742.1			
	341.5					1312.6			
R112	970.5	63.1							
	1768.9								
	318.4								

Table 10: Results of instance from Solomon (1986). First value represents the distance, second value the total scheduling time and third the waiting time. The difference with the optimal solutions given in Kallehauge et al. (2005) is computed. Times of solutions given in Kallehauge et al. (2005) were not given.

			Kallehauge et al					Kallehauge et al	
Instance	KGLS	Time	Solution	Gap	Instance	KGLS	Time	Solution	Gap
RC101	1665.7	28.7	1619.8	2.83%	RC201	1290.4	165.3	1261.8	2.26%
	2158.8					5685.5			
	1133.4					5259.2			
RC102	1490.1	152.9	1457.4	2.25%	RC202	1112.3	154.5	1092.3	1.83%
	2222.2					5053.0			
	1259.8					4376.1			
RC103	1313.7	51.6	1258	4.43%	RC203	951.3	94.0		
	2097.1					3849.1			
	813.3					2692.6			
RC104	1172.1	153.8			RC204	806.2	82.9		
	2028.7					3205.7			
	490.8					1862.2			
RC105	1581.8	30.6	1513.7	4.50%	RC205	1181.0	33.1	1154	2.34%
	2019.2					5907.7			
	1548.4					5218.7			
RC106	1412.4	121.3			RC206	1085.6	84.4		
	2059.1					4094.2			
	492.0					3265.0			
RC107	1230.4	102.6	1207.8	1.87%	RC207	980.3	53.1		
	1882.3					3445.0			
	440.6					2337.6			
RC108	1140.8	93.9	1114.2	2.39%	RC208	794.1	147.6		
	1730.5					2885.4			
	321.6					1578.1			

9.3 VRP heterogeneous fleet

Table 11: Results from the instances given in Golden et al. (1984). The gap is reported in % and between the solution of KGLS and the average solution from Taillard et al. (1997).

	KGLS		Taillard et al			
	Value	Time	Best	Average	Time	Gap
13 (50)	2503.9	50	2413.8	2436.8	470	2.75%
14 (50)	9648.1	11	9119.0	9123.6	570	5.75%
15 (50)	2600.9	39	2586.4	2593.6	334	0.28%
16 (50)	2807.4	81	2741.5	2744.3	349	2.30%
17 (75)	1814.7	46	1747.2	1753.7	2072	3.48%
18 (75)	2392.4	47	2373.6	2382.8	2744	0.40%
19 (100)	8670.2	35	8661.8	8665.4	12528	0.05%
20 (100)	4164.6	148	4047.6	4063.2	2117	2.50%
		57			2648	2.19%

9.4 Pseudo-code

1. Construct a starting solution with CW, if $M_{CW} > M_{min} + 1$, use CW with adapted savings.

2. Optimise individual routes with LK.

3.Apply CE and RC. If a route is changed re-optimise it with LK, set b.

4. While time < time limit

  5. While P<30

    6. Penalize edge (i,j) with highest b(i,j) value by increment p(i,j).

    7. Use CE and RC on (i,j) using c^g

  end while

  8. Apply LK, CE and RC on all changed routes with normal c.

  If a route is changed with CE or RC re-optimise it. Change b.

end while