ConRTR, a Consistent Vehicle Routing Problem algorithm

Rick den Ridder

June 30, 2014

Abstract

Consistent vehicle routing problems (ConVRP) are VRP's with multiple days, where customers always need to be visited at roughly the same time of the day. To solve these problems, Groër et al. (2009) developed an algorithm called ConRTR. In this study, I will show that some of the solutions as provided by Groër et al. (2009) are infeasible and how ConRTR can be used to get feasible solutions. It is studied how all parameters of ConRTR influence the solution and computation time. will show what the influence is of all used parameters in ConRTR on the solution and computation time. I will also provide an algorithm that makes solutions more consistent. With a relative small increase in total travel time, the ConRTR solutions can become much more consistent.

1. Introduction

Vehicle routing problems (VRP's) are problems in which the total costs, total distance or total travel time is minimized, while certain constraints need to be met. These constraints can, among other things, relate to the vehicle capacity and time. VRP's can be used in the package delivery industry, where multiple customers receive at least one package on one or more days. The delivery company plans routes for all days, deciding which vehicles will visit which customers in which order.

VRP's are a subject of multiple studies. Groër et al. (2009) studied a Consistent Vehicle Routing Problem (ConVRP). This is a VRP in which the total travel time is minimized, while meeting the capacity and maximum travel time constraints. Furthermore, additional constraints are added to make sure that if customers are visited on more than one day, they are always visited by the same driver and at roughly the same time of the day. In this ConVRP, there are no time window constraints stating certain customers should be visited within a certain time window. Besides, there is no constraint on the maximum number of vehicles. When a customer is visited on multiple days, the difference between the latest and earliest arrival time at his address is called the arrival time differential.

The ConVRP can be modeled as a Mixed Integer Program (MIP). In practice, this MIP can only be used to solve small instances of the ConVRP. For larger problems, the computation time will be too large to solve the problem in time. To handle large problems, Groër et al. (2009) developed an algorithm that uses the record-to-record (RTR) travel algorithm of Li et al. (2005) and the modified Clarke and Wright algorithm with parameter λ of Yellow (1970). It is called ConVRP RTR travel (ConRTR). ConRTR does not contain a hard constraint on the arrival time differential. It assumes that the algorithm itself provides consistent solutions. ConRTR is built around the precedence principle; if customer a receives a package before customer b on a certain day, the algorithm makes sure that customer b will not receive a package before customer a on another day.

The ConRTR algorithm splits all customers into two groups. Customers that are visited on more than one day are placed in group G_1, all other customers are placed in group G_2. Then, template routes for the customers in group G_1 are generated. For every day, this template is adjusted for those customers in group G_2 who are only visited on that specific day and for those customers in group G_1 who are not visited on that specific day. When only a few adjustments are needed every day, the arrival times will be similar on all days and the algorithm will perform better.

Because ConRTR minimizes the total travel time and does not contain a hard constraint on the arrival time differentials, vehicles will never wait between visiting two customers. If this would be allowed, solutions with a lower arrival time differential may be possible. However, this will lead to a larger total travel time.

Groër et al. (2009) tested their algorithm on multiple problems. For small problems, the solutions of the ConRTR algorithm are compared with optimal solutions of the Mixed Integer Problem. To make a fair comparison, they distinguished solutions with a maximum time differential of five and solutions without a maximum time differential. For larger problems, including modified benchmark problems, the solutions are compared with solutions of the RTR algorithm. The RTR algorithm generates solutions within 1% to 2% of the best known solutions, as stated in Li et al. (2005). Finally, the algorithm is applied to real world data. With this real world dataset, it is also concluded that it is possible to make well performing template routes using only historical data.

In this paper, I will try to verify the results of the ConRTR as stated in Groër et al. (2009). Furthermore, I will show how the solutions and computation times depend on the used parameters. Finally, I will try to decrease the maximum arrival time differential by allowing waiting times and I will study how much this increases the total travel time.

2. The ConVRP

The ConVRP is a VRP for D days and N customers. To solve the ConVRP, the following parameters are introduced.

- Q, the capacity of the vehicles.
- T, the maximum travel time per vehicle per day.
- L, the maximum arrival time differential.
- The demand of all N customers on all D days.
- The service time for all N customers on all D days.
- Travel times between all combinations of locations (the depot and the N customers).

A solution of the ConVRP is a set of routes. A route contains information about the trip of one vehicle on one day. It specifies which customers are visited by a vehicle on a specific day and on what time the vehicle arrives and departs at the depot and all visited customers.

A solution is feasible when all of the following constraints are met.

- Every day, all customers receive the demanded amount of packages.
- A customer is only visited once per day.
- A vehicle cannot transfer more packages than its capacity Q per day.
- The total travel time of a vehicle on one day cannot exceed the maximum travel time T.
- If a customer is visited on more than one day, it is always visited by the same vehicle.
- Every route starts and ends at the depot.
- For every customer, the arrival time differential must be less than L.

The optimal solution is the feasible solution with the smallest total travel time, including service and waiting times.

3. The ConRTR algorithm

In this section, I will explain the ConRTR algorithm and show which parameters can be changed to obtain better solutions or to decrease the computation time.

The first step of ConRTR is to divide the N customers into two groups. Group G_1 contains the customers that need to be visited on more than one day, all other customers are in group G_2. With the customers in group G_1, template routes are generated. For every day, these template routes need to be adjusted. There could be customers in group G_1 that don't need to be visited on a specific day. These customers will be deleted from the template routes in a later phase. There could also be customers in group G_2, who are not in the template routes, that need to be visited on a specific day. These customers will be added to the template routes in a later phase. Due to this additions and deletions, the actual routes can become infeasible or far from optimal. When more customers need to be added than deleted, the template routes need to have a buffer to deliver packages to those extra customers. On the other hand, when more customers are going to be deleted than added, it is allowed to overload the template routes a little bit. To account for this later adjustments, the template routes are generated with an adjusted capacity and maximum total travel time constraint.

$$Q_{template} = Q \times \left(\frac{number\ of\ customers\ in\ G_1}{average\ number\ of\ stops\ per\ day} \right)$$

The amount of packages to deliver depends linearly on the number of customers. If more customers are added than deleted, the average number of stops per day will be larger than the number of customers in G_1. $Q_{template}$ will therefore be lower than Q, which results in template routes that have a buffer for extra customers.

$$T_{template} = T \times \sqrt{\frac{number\ of\ customers\ in\ G_1}{average\ number\ of\ stops\ per\ day}}$$

The total travel time will not increase or decrease linearly when a customer is added or deleted. For example, if two neighbors need to be visited on a day, the total travel time will be almost equal to the total travel time of the route where only one of those neighbors needs to be visited. For this reason, the square root of the fraction $\frac{number\ of\ customers\ in\ G_1}{average\ number\ of\ stops\ per\ day}$ is used in the formula above. Groër et al. (2009) use a slightly different formula, where T is divided by the square root of this fraction. When more customers need to be deleted than added, their formula decreases $T_{template}$ and thereby create a buffer. I don't quite follow this reasoning, so I will use the formula as stated above.

Furthermore, for all customers in the template, the number of packages to receive and the service times are set to the mean amount for every of those customers while template routes are generated.

Initial template routes are created with the modified Clarke and Wright algorithm and a parameter λ as stated in Yellow (1970). The modified Clarke and Wright algorithm, also known as the savings algorithm, merges two routes when this combination yields the largest savings and all constraints are still met. When a route ending with customer a and a route starting with customer b are merged, the savings S_{ab} are defined as

$$S_{ab} = d_{a0} + d_{0b} - \lambda \cdot d_{ab}$$

where

d_{ab} = distance from customer a to customer b
d_{a0} = distance from customer a to the depot
λ = route shape parameter.

Savings are larger when customers are further away from the depot or when the distance between the two customers is small. A smaller value for λ will increase the influence of the distance between the customers and the depot on the savings. It will make it more likely to connect two customers that are far from the depot, even if the distance between those two customers is not very small. A larger value for λ will have the opposite effect. It will make it more likely to connect two customers that are close together, even if both are close to the depot. This is illustrated in the example below.

Set $d_{a0} = 10$, $d_{b0} = 10$, $d_{ab} = 5$, $d_{c0} = 6$, $d_{d0} = 6$ and $d_{cd} = 2$. With $\lambda = 1$, $S_{ab} = 15$ and $S_{cd} = 10$. Customer a and b are connected because they are far away from the depot, even though the distance between customer c and d is smaller than the distance between customer a and b. With $\lambda = 3$, $S_{ab} = 5$ and $S_{cd} = 6$. Customer c and d are connected because they are close together, even though customer a and b are further away from the depot than customer c and d. In this example, customer a and b will be connected if $\lambda < 2\frac{2}{3}$ and customer c and d will be connected if $\lambda > 2\frac{2}{3}$. If $\lambda = 2\frac{2}{3}$, the savings are equal.

The template routes contain all customers in group G_1 and will be used to make the actual routes for all days. This means that for every day, the template routes are being adjusted. The customers that are in group G_1 but don't have to be visited on that day, are deleted from the routes. For example, I take a template route *a*-*b*-*c*. If customer b does not need to be visited on a day, the adjusted template route becomes *a*-*c*. If no customers are added later, all customers that are visited after customer a will have an earlier arrival time than in the template routes.

The customers that are in group G_2 and only need to be visited on that day, are added to the routes. When customer d is added in the previous example, there are three possible routes: *d*-*a*-*c*, *a*-*d*-*c* and *a*-*c*-*d*. It is not possible that customer c will be visited before customer a in the adjusted template route, so the precedence principle holds.

Groër et al. (2009) don't state exactly how the customers from group G_2 are added. In the algorithm that I used, the savings are calculated for every customer that needs to be added and for every position in the routes. These savings are the time differences between the total travel time of the adjusted route with the extra customer, and the sum of the total travel time of a route containing only the extra customer and the total travel time of the template route. For example, when customers *d* and *e* are added to template route *a*-*b*-*c*, the savings for customer *d* on position one is the travel time of *a*-*b*-*c* plus the travel time of *d* minus the travel time of *d*-*a*-*b*-*c*. The savings for customer *e* on position three is the travel time if *a*-*b*-*c* plus the travel time of *e* minus the travel time of *a*-*b*-*e*-*c*.

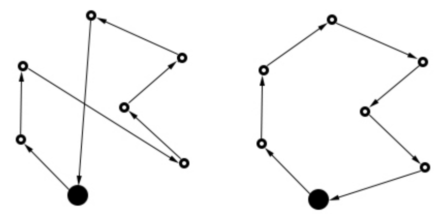
It is also checked if an adjusted route is still feasible. If the combination of an extra customer and a position in the routes gives the largest savings and it will not lead to infeasible routes, this adjustment is made. After that, savings are calculated again for the customers that still need to be added and this process is repeated until all customers are added.

If it is not possible to add a customer and preserve feasible routes, new initial template routes are generated. These are generated with a lower $Q_{template}$ and/or $T_{template}$. These values decrease with $\frac{mean\ violation}{2}$ until a feasible solution is generated. The template routes, actual routes for all days and the total travel time of the actual routes are saved.

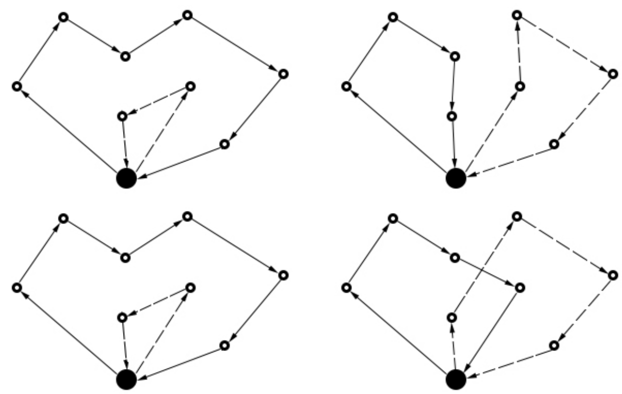
When a feasible solution is obtained, a diversification phase is started. To change the template routes, three operations of Li et al. (2005) can be performed: one-point move, two-point move and two-opt move. A one-point move is the move of one customer to another place in the same route or in another route. With a two-point move, two customers switch position in the routes. With a two-opt move, two direct connections between two customers are removed. If these two connections belong to the same route, there are two ways to reconnect the customers. One is to connect them as they were, the other will result in a different route. When performing a two-opt move within a route, the reconnection that results in the same route will not be made. When the two connections belong to different routes, there are three ways to reconnect the customers. Two of those three reconnections will actually change the routes. From those two reconnections, the one that results in routes with the lowest travel time is used when performing a two-opt move between routes. Figure 1 contains examples of two-opt moves.

Figure 1 Examples of two-opt moves

(a) Two-opt move within a route



(b) Two-opt moves between two routes



In the diversification phase, moves on the template routes are allowed when all constraints are satisfied and the total travel time does not increase more than a certain deviation. This deviation is set to the total travel time of the initial template routes multiplied by $(1 + \alpha)$. ConRTR from Groër et al. (2009) applies *I* operations in this phase, but it is not clear which moves are performed in which order. The algorithm that I use will apply random moves until *I* moves are accepted. To select a random move, first the type of move (one-point, two-point or two-opt) is chosen randomly with equal probabilities. Afther that, the specifications (customers, positions and edges) of the move are chosen randomly with equal probabilities. Because of this random element in the algorithm, the results can differ when the algorithm is used multiple times on the same problem.

After the *I* random operations, the algorithm will try to improve the template routes during an improvement phase. This is done by applying one-point moves, two-point moves and two-opt moves that result in feasible routes with a lower total travel time, until there are no more improving moves possible. In the algorithm that I use, all improving one-point moves are performed first, followed by all improving two-point moves and all improving two-opt moves. If there was at least one improving two-point or two-opt move, all moves are tried again. The pseudo code for this phase is stated in Figure 2.

Figure 2 Pseudo code of the improvement phase

```
set two-point-boolean = 0
set two-opt-boolean = 0

while two-point-boolean = 0 and two-opt-boolean = 0

    set two-point-boolean = 0
    set two-opt-boolean = 0

    for all customers
        for all positions
            try one-point move with customer and position
            if total travel time has decreased and new routes are feasible
                accept move
            end
        end
    end

    for all customers as customer1
        for all customers as customer2
            try two-point move with customer1 and customer2
            if total travel time has decreased and new routes are feasible
                accept move
                set two-point-boolean = 1
            end
        end
    end

    for all edges as edge1
        for all edges edge2
            try two-opt move with edge1 and edge2
            if total travel time has decreased and new routes are feasible
                accept move
                set two-opt-boolean = 1
            end
        end
    end
end
```

When the template routes cannot be improved any more, they will be used to make the actual routes for all days. Customers will be deleted from and added to the template routes as explained before. If this results in infeasible routes, $Q_{template}$ and/or $T_{template}$ are decreased with $\frac{mean\,violation}{2}$ and the improvement phase will start over. If there are no feasible routes after at most K decreases, the improvement phase is stopped and the last known feasible template is used for the next step of the algorithm. If the actual routes are feasible, $Q_{template}$ and $T_{template}$ are being increased with $\frac{mean\,slack}{2}$ and the next phase is started.

After the improvement phase, the total travel time of the actual routes is calculated. When this is lower than any previous total travel time, the template routes, actual routes and total travel time are saved. The diversification phase and improvement phase are repeated with the adjusted $Q_{template}$ and $T_{template}$ until this results in J consecutive actual routes with a total travel time that is not lower than any previous total travel time.

ConRTR repeats all the previous steps with different values for λ. After that, the feasible solution with the lowest total travel time is returned as the ConRTR solution.

The ConRTR algorithm uses five parameters that can be changed. A change can result in solutions with a smaller or larger total travel time and a larger of smaller computation time. I will state which parameters Groër et al. (2009) used and what the potential effects of a parameter change are.

- λ is 0.6, 1.0 or 1.4. When running ConRTR with a larger vector of lambda's, it may be possible to obtain solutions with a lower total travel time. The computation time will increase in that case, as all steps of ConRTR are repeated a larger number of times.
- I is set to 30. A larger value can result in more diversification, but it is not clear if that leads to solutions with a lower total travel time.
- α is set to 0.01. A larger value can result in more diversification, but it is not clear if that leads to solutions with a lower total travel time.
- K is set to 5. A larger value results in a larger computation time, because the template routes are adjusted more often when they result in infeasible actual routes. However, it may generate solutions with a lower total travel time. For a lower value, the opposite holds.
- J is set to 5. A larger value results in a larger computation time, because the improvement phase is repeated more often. However, it may generate solutions with a lower total travel time. For a lower value, the opposite holds.

4. Reducing the maximum arrival time differential

ConRTR makes use of the precedence principle to obtain consistent solutions, but it does not minimize the arrival time differential. Because the total travel time is minimized, vehicles will never wait during their routes. If waiting is allowed, solutions with a lower arrival time differential may be possible. The total travel time will of course increase, but it could be that a relative small increase in total travel time leads to a relative large decrease of the maximum arrival time differential. This could be useful when drivers are paid per day instead of per hour. It makes waiting time relatively cheap, while a smaller maximum arrival time differential would lead to a larger customer satisfaction level.

I will describe an algorithm that reduces the maximum arrival time differential of a ConVRP solution. It can be applied after the ConRTR algorithm is performed.

For all customers in group G_1, the difference between the earliest and latest arrival time is calculated. The customer with the largest arrival time differential is taken and the record differential is set to this differential. If the maximum arrival time differential is found at multiple customers, the customer who is visited first in the template routes is taken. Equal arrival time differentials occur when customers are visited directly after each other on all days that they are visited. For this customer, it is determined on which days it is visited and what the arrival times on those days are. The day with the earliest arrival time is taken.

It is tried to let the vehicle wait before it visits the customer on this day. The waiting time is set to the difference between the maximum arrival time differential and the second to largest arrival time differential + 0.1. If the maximum arrival time differential is not unique, the second to largest arrival time differential is the largest arrival time differential that is not equal to the maximum arrival time differential. If the maximum arrival time differential is smaller than 0.2, the waiting time is set to the maximum arrival time differential. Setting the waiting time like this will decrease the maximum arrival time differential with the smallest possible increase in total travel time.

It is then determined what the longest possible waiting time is. This is the maximum travel time of a vehicle minus the travel time of the complete route for the vehicle. If the longest possible waiting time is smaller than the waiting time, the waiting time is set to the longest possible waiting time.

The arrival time of this customer and all customers that are visited after it are increased by the waiting time. All steps are repeated with the new time scheme until the maximum arrival time differential is zero or if it is not possible to let a vehicle wait before visiting the customer with the maximum arrival time difference on the day with the earliest arrival time.

5. Computational results

The results of this study are stated in this section. First, I try to verify the results stated in Groër et al. (2009). After that, I study the effect of parameter changes. The results of the these changes are stated in section 5.3 and further. In section 5.7, the results with a reduced maximum arrival time differential are stated.

Because there is a random element in ConRTR, I will run ConRTR five times for every instance. There are also five seeds for the random number generator that will be used in the experiments. This is necessary to get different results in the five runs, while it is still possible to compare the results of experiments with different parameters. The reported computation time is the mean of the five runs. The minimum, maximum and mean total travel time of the five runs are also reported, as well as the maximum arrival time differential belonging to the solution with the minimum total travel time.

ConRTR is performed in Matlab, while using a 2.4 GHz Intel i7 processor and 8 GB of RAM.

5.1. Problem instances

When deciding which problem instances will be used, the following characteristics are important: the number of customers, the locations of the customers and the location of the depot.

The location of the customers and the depot can influence the performance of ConRTR. It can be interesting to know how ConRTR performs with clustered and unclustered customers. The location of the depot can vary from the middle of the area with customers to an edge of that area.

A larger number of customer will be more representative for real world problems, but it will also result in larger computation times. Preliminary experiments showed that the computation time is a few seconds for VRP's with 10 customers and at most five days. The computation time increases to around a minute when the number of customers in the VRP increases to 25 customers. An increase to 50 or 100 customers results in computation times of several minutes and around 20 minutes respectively.

To study the effects of parameter changes, I use four different problems. They all contain five days and 25 customers, as larger problems result in computation times that I consider too long to do all the experiments. Problem 1 and problem 3 have clustered customers, while problem 2 and problem 4 have unclustered customers. The locations of the unclustered customers are randomly and uniformly generated in a square area. The locations of the clustered customers are randomly and uniformly generated in three smaller square areas. These three areas are randomly and uniformly placed in the total square area. Problem 1 and problem 2 have a depot in the middle of the area, while problem 3 and problem 4 have a depot on a vertex of the area. The constraints, demands and service times are the same for all problems. The problem instances are stated in Appendix A.

5.2. Results of Groër et al. (2009)

Groër et al. (2009) performed ConRTR on multiple ConVRP's. The number of customers varied between 10 and 1,000. For 22 problems, the problem instances are available. For 10 of those 22 problems, the optimal solutions were provided online. These are small problems with 10 or 12 customers and three days.

Some of the optimal solutions are not feasible. For example, the ConRTR solution with arrival times at all customers and the depot (0) of the first small problem is stated in Table 1. If the arrival times of the vehicles at the depot are added, the total travel time of 142.03 as stated in Groër et al. (2009) is confirmed. The maximum travel time T of this VRP is 30. However, it can be seen that vehicle 2 always arrives later at the depot. This makes this solution infeasible.

It can be that ConRTR from Groër et al. (2009) don't consider service times when calculating the travel time of one vehicle, even though this is included in their formulation of the ConVRP and in their calculations for the total travel time. To check this, I ran my version of ConRTR two times for the 10 problems. Once with and once without considering service times when checking the feasibility of the routes. These results are stated in Table 2.

Table 1 Solution of a small ConVRP

	Day 1		Day 2		Day 3	
	Cust.	Arrival time	Cust.	Arrival time	Cust.	Arrival time
Vehicle 1	0	0	0	0	0	0
	0	0	9	9.45	9	9.45
			7	17.68	0	19.90
			0	22.06		
Vehicle 2	0	0	0	0	0	0
	10	2.98	6	11.59	10	2.98
	3	5.93	1	14.36	3	5.93
	1	15.23	5	20.64	6	13.95
	5	21.51	8	24.91	1	16.72
	8	25.78	2	28.52	8	25.61
	4	30.09	4	30.24	0	32.51
	0	33.71	0	33.85		

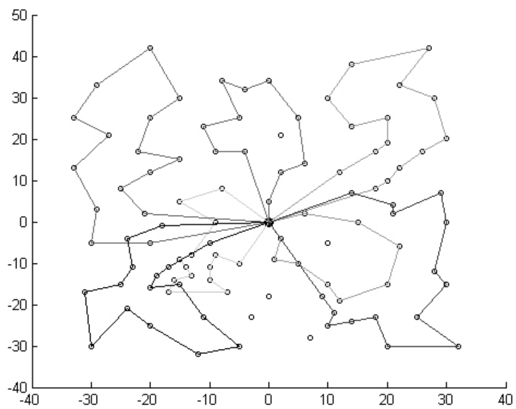
Table 2 ConRTR for small problems

Groër et al.		ConRTR service		ConRTR no service		
Time	Max diff.	Time	Max diff.	Time	Max diff.	Gap
142.03	3.70	142.03	3.70	184.76	7.23	30%
121.07	4.40	127.06	4.57	131.41	1.27	3%
149.41	3.48	149.41	3.48	149.41	3.48	0%
150.89	2.87	144.07	2.87	151.77	2.87	5%
130.77	12.37	130.77	12.38	130.77	12.38	0%
171.05	3.96	171.32	3.72	199.94	5.18	17%
111.54	4.25	116.98	14.54	121.12	19.87	4%
145.67	5.26	142.97	10.44	160.28	3.85	12%
165.23	5.47	165.23	6.49	192.19	6.84	16%
140.29	7.90	140.29	7.90	147.17	7.22	5%

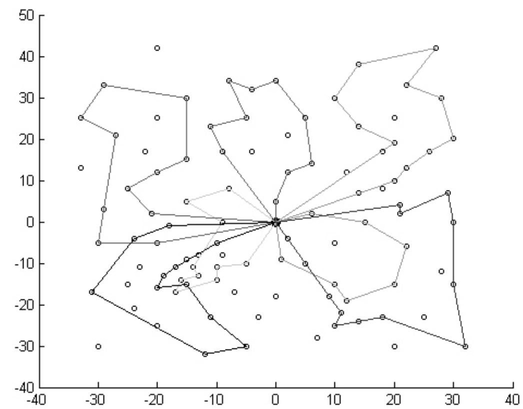
It can be seen that with five of the 10 instances, ConRTR without considering service times finds the same solution as ConRTR from Groër et al (2009). The differences for the other five instances can arise due the diversification phase or the addition of customers from group G_2, because these where not stated clear in Groër et al (2009). ConRTR with service times considered finds worse solutions in eight of the 10 instances and equal solutions with the other two instances. It seems plausible to conclude that ConRTR from Groër et al (2009) does not consider service times when checking the feasibility of routes. The difference between the total travel times of the solutions found with and without considering service times is stated as 'Gap' and varies between 0% and 30%.

I also performed ConRTR for a larger problem. This contains 100 customers and five days. To see if ConRTR performs as expected, I showed a plot of the template and actual routes in Figure 3.

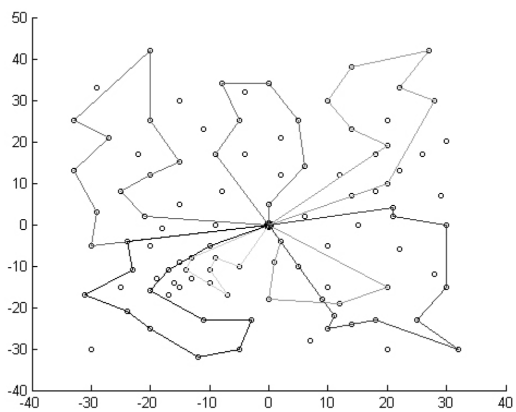
Figure 3 ConRTR solution for problem with 100 customers and five days



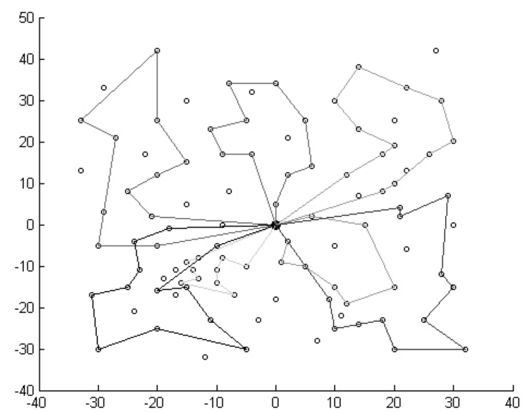
(a) Template



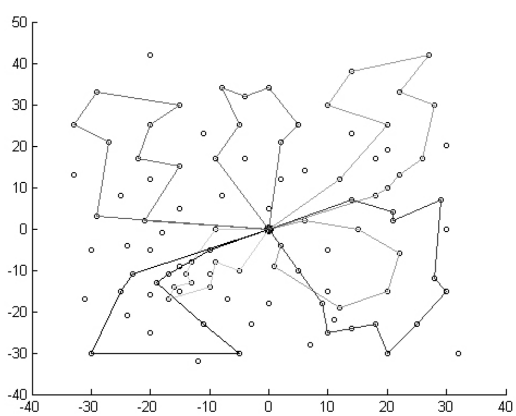
(b) Day 1



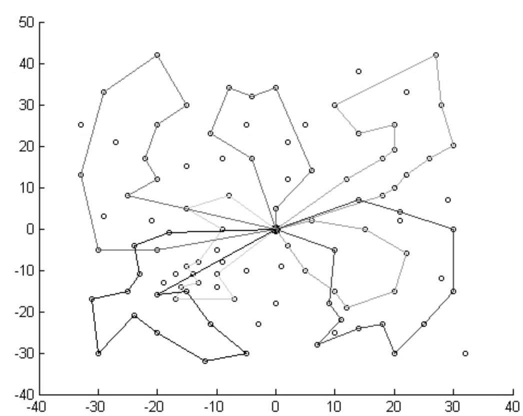
(c) Day 2



(d) Day 3



(e) Day 4



(f) Day 5

5.3. The modified Clarke and Wright algorithm

As explained in chapter 3, ConRTR makes use of the modified Clarke and Wright algorithm with parameter λ. The ConRTR algorithm from Groër et al. (2009) has a vector with three values for λ: 0.6, 1 and 1.4. To check if it is possible to obtain better results with different values for λ, I expanded the vector. It contains the values 0.3, 0.6, 0.8, 1, 1.2, 1.4, 2 and 4. The results of ConRTR with this expanded vector are in Table 3.

There is only one problem where the expanded vector gives a better solution than the original. This is at problem 1, where the solution has a total travel time that is 4.6% lower than found before. This also increases the computation time, it gets twice as long. If better solutions are very important and computation time is not an issue, the vector with values for λ can be expanded. Based on this data, it cannot be concluded that deleting the value 1.4 from the vector of values for λ will not lead to worse solutions. With λ set to 1.4, solutions are obtained that are quite close to the solutions with value 0.6 or 1. It could be that λ set to 1.4 give better solutions for other problems.

5.4. The diversification phase

There are two parameters that influence the diversification phase: *I* and α. *I* determines how many moves are performed in this phase and α determines which moves are allowed.

Table 4 shows the results of ConRTR when the parameter *I* is varied. All other parameters are constant and equal to the parameters used in the ConRTR algorithm from Groër et al. (2009). If *I* is set to 0, the diversification phase is skipped. For all problems, better solutions can be obtained with a diversification phase. In almost all cases, even the solution with the largest total travel time is better than or equal to the solution found without a diversification phase. This makes this phase highly recommended. It is not clear what the optimal value for *I* is. For problem 4, the best result was obtained with *I* set to 15, for problem 1 and 2, it was obtained with *I* set to 60.

As the moves in this phase are randomly performed, it could be a matter of luck when it comes to obtaining better solutions. If the last move is the one that makes a better solution possible, it does not matter if it was move 15 out of 15 or 60 out of 60. The fact that it is always better to have a diversification phase, can be explained by the fact that this phase is performed multiple times in the ConRTR algorithm. There is a large probability that one of those times, a random move turns out to be good at the right moment.

The computation time does not increase much when *I* is increased, so that should not be a factor when deciding which value for *I* need to be used.

Table 4 Results for ConRTR with different values for *I*

Problem	*I*	Minimum total travel time	Maximum total travel time	Mean total travel time	Maximum differential	Computation time
1	0	1,133.79	1,133.79	1,133.79	28.19	28.74
1	15	1,035.45	1,133.79	1,056.64	10.22	31.30
1	30	1,034.64	1,040.89	1,038.13	8.38	32.04
1	60	1,028.52	1,087.68	1,046.15	13.74	48.04
2	0	1,172.78	1,172.78	1,172.78	22.53	27.70
2	15	1,125.65	1,139.62	1,132.80	17.79	27.99
2	30	1,119.62	1,156.57	1,137.01	17.79	26.61
2	60	1,110.13	1,148.00	1,129.50	9.20	32.76
3	0	919.91	919.91	919.91	10.15	23.14
3	15	919.70	919.91	919.87	12.76	32.89
3	30	919.70	919.91	919.87	12.76	33.80
3	60	919.91	919.91	919.91	10.06	32.08
4	0	1,857.59	1,857.59	1,857.59	20.01	60.82
4	15	1,555.41	1,887.70	1,811.02	16.56	45.41
4	30	1,746.78	1,887.70	1,804.98	31.80	38.87
4	60	1,756.29	1,857.59	1,811.14	27.54	50.04

Table 5 shows the results of ConRTR when the parameter α is varied. All other parameters are constant and equal to the parameters used in the ConRTR algorithm from Groër et al. (2009). If α is set to 0, deteriorating moves are not allowed in the diversification phase. It is not recommended to do this, because the phase lasts until *I* moves are accepted. With α set to 0 and *I* larger than 0, it could be that no move is accepted and ConRTR stays forever in this phase. To run ConRTR without deteriorating moves, the diversification phase can be skipped by setting *I* to zero.

There is no clear correlation between the value of α and the total travel time of the best solution. A small value for α has the advantage that template routes that are far away from the optimal routes are not created. However, it could be possible that the optimal solution of actual routes requires template routes that are not optimal. A small value for α might prevent the creation of such template routes. A large value for α has the opposite advantages and disadvantages. The computation time tends to increase when α is large. This is probably because more routes with a larger total travel time are created and it therefore takes longer to get to the ConRTR solution.

Table 3 Results for ConRTR with expanded vector for λ

Problem	Vector	λ	Minimum total travel time	Maximum total travel time	Mean total travel time	Maximum differential	Computation time
1	original	0.6	1,034.64	1,040.89	1,038.13	8.38	32.04
1	expanded	0.8	986.64	1,036.29	1,025.04	9.90	66.47
2	original	0.6	1,119.62	1,156.57	1,137.01	17.79	26.61
2	expanded	0.6	1,119.62	1,156.57	1,137.01	39.67	39.67
3	original	1	919.70	919.91	919.87	12.76	33.80
3	expanded	1	919.70	919.91	919.87	12.76	56.39
4	original	0.6	1,746.78	1,887.70	1,804.98	31.80	38.87
4	expanded	0.6	1,746.78	1,842.95	1,789.78	31.80	53.33

Table 5 Results for ConRTR with different value for α

Problem	α	Minimum total travel time	Maximum total travel time	Mean total travel time	Maximum differential	Computation time
1	1%	1,034.64	1,040.89	1,038.13	8.38	32.04
1	10%	1,037.54	1,037.98	1,037.89	11.49	42.07
1	20%	1,033.31	1,037.98	1,037.05	9.90	44.29
2	1%	1,119.62	1,156.57	1,137.01	17.79	26.61
2	10%	1,127.67	1,139.45	1,132.47	17.79	31.60
2	20%	1,148.63	1,174.67	1,162.42	17.13	34.54
3	1%	919.70	919.91	919.87	12.76	33.80
3	10%	913.43	919.91	918.57	37.94	40.47
3	20%	919.91	919.91	919.91	10.06	40.07
4	1%	1,746.78	1,887.70	1,804.98	31.80	38.87
4	10%	1,755.74	1,881.13	1,805.23	33.36	68.04
4	20%	1,713.19	1,857.44	1,804.08	22.82	55.32

Table 6 Results for ConRTR with different values for *K*

Problem	K	Minimum total travel time	Maximum total travel time	Mean total travel time	Maximum differential	Computation time
1	0	1,034.64	1,038.58	1,037.67	8.38	17.19
1	3	1,017.91	1,038.58	1,034.33	40.76	31.01
1	5	1,034.64	1,040.89	1,038.13	8.38	32.04
1	10	1,030.55	1,040.89	1,037.32	9.90	61.12
2	0	1,119.62	1,154.33	1,136.57	17.79	15.64
2	3	1,119.62	1,154.33	1,136.57	17.79	22.08
2	5	1,119.62	1,156.57	1,137.01	17.79	26.61
2	10	1,119.62	1,156.37	1,137.01	17.79	42.38
3	0	919.91	919.91	919.91	10.06	17.25
3	3	919.70	919.91	919.87	12.76	28.87
3	5	919.70	919.91	919.87	12.76	33.80
3	10	911.79	919.91	918.29	25.20	43.69
4	0	1,836.56	1,887.70	1,871.45	16.19	24.66
4	3	1,775.36	1,887.70	1,865.23	15.93	40.50
4	5	1,746.78	1,887.70	1,804.98	31.80	38.87
4	10	1,822.44	1,887.70	1,853.41	19.79	66.25

5.5. The improvement phase

In the improvement phase, the template routes are adjusted. After these adjustments, customers are deleted and added to make the actual routes. If the actual routes are infeasible, the capacity and maximum total travel time constraints are adjusted. Then, the improvement phase is repeated until feasible routes are created or the constraints are adjusted K times.

Table 6 shows the results of ConRTR when the parameter K is varied. All other parameters are constant and equal to the parameters used in the ConRTR algorithm from Groër et al. (2009). It was expected that a larger value for K would lead to better solutions, but even though the seed of the random number generator is set the same for every instance, this is not always the case. This can be explained by the fact that a different value for K will change the number of feasible routes found. When more feasible routes are found, the diversification phase is repeated more often. If this happens before the best solution is generated, the random number generator is in a different state and the diversification phase might not adjust template routes as needed for the best solution.

With K larger than zero, the solutions are at least as good as with K set to zero. It is therefore recommended to set K larger than zero, even though this increases the computation time. With K set to 10, the computation time is 60% longer than with K set to 5. The computation times with K set to 3 or 5 are quite similar. Because it cannot be concluded that better solutions are obtained with K set to 10, it is recommended to set K to 3 or 5. It is not clear which one of those two gives better solutions.

5.6. Repetition of diversification and improvement

The diversification and improvement phase are repeated until J consecutive repetitions do not lead to a better solution.

Table 7 shows the results of ConRTR when the parameter J is varied. All other parameters are constant and equal to the parameters used in the ConRTR algorithm from Groër et al. (2009). It was expected that a larger

Table 7 **Results for ConRTR with different values for *J***

Problem	J	Minimum total travel time	Maximum total travel time	Mean total travel time	Maximum differential	Computation time
1	0	1,034.64	1,133.79	1,057.06	8.38	13.45
1	3	1,034.64	1,038.58	1,037.34	8.38	27.69
1	5	1,034.64	1,040.89	1,038.13	8.38	32.04
1	10	1,034.64	1,038.58	1,037.46	8.38	71.42
2	0	1,119.62	1,156.57	1,137.01	17.79	8.90
2	3	1,119.62	1,154.33	1,136.57	17.79	19.94
2	5	1,119.62	1,156.57	1,137.01	17.79	26.61
2	10	1,119.62	1,156.57	1,137.01	17.79	52.95
3	0	919.50	919.91	919.79	10.15	8.09
3	3	913.43	919.91	918.62	5.26	24.46
3	5	919.70	919.91	919.87	12.76	33.80
3	10	919.91	919.91	919.91	10.15	66.59
4	0	1,858.43	2,122.82	1,936.51	18.35	9.99
4	3	1,756.29	1,921.72	1,842.85	27.54	27.34
4	5	1,746.78	1,887.70	1,804.98	31.80	38.87
4	10	1,745.67	1,887.70	1,782.59	31.80	61.70

value for *J* would lead to better solutions, but even though the seed of the random number generator is set the same for every instance, this is not always the case. This can be explained by the fact that a different value for *J* will change the number of times the diversification phase is repeated. If this happens with the first value for λ, the random number generator is in a different state when the first diversification phase with the second value for λ is started. This can lead to different solutions.

In general, the mean total travel time of the solutions decrease when *J* is increased. It makes a solution with a large total travel time less likely and it is therefore recommended to set *J* to a value larger than zero. A large value gives a better solution, but it also increases the computation times. Because the computation times with *J* set to 5 are quite similar to the compuation times with *J* set to 3 and about half as long as with *J* set to 10, it is recommended to set *J* to 3 or 5. It is not clear which one of those two gives better solutions.

5.7. Decreasing the arrival time differential

In section 4, I described an algorithm to decrease the arrival time differential. This algorithm is applied to five problems. The four problems as used in the previous experiments and a larger problem with 100 customers.

The results of this algorithm are stated in Table 8. For the small problems, it was possible to visit all customers at the exact same time on all days. This will increase the total travel time with at most 10%. For the large problem with 100 customers, it was not possible to get a maximum arrival time differential of zero. However, the maximum arrival time differential could still be decreased with more than 90%, while increasing the total travel time with less than 10%. If the increase of total travel time would be limited to 5%, the maximum arrival time differential could still improve with 80%.

For the first four problems, the computation time is less than a second. For the large problem with 100 customers, the computation time is 10 seconds. The computation time can be decreased if the maximum arrival time differential is being decreased in larger steps. I used a waiting time that was 0.1 larger than the difference between the two largest arrival time differentials. If instead of 0.1, 0.5 would be used, the computation time for the large problem becomes less than two seconds. However, this leads to less consistent results, as the maximum arrival time differential would decrease to 3.57 instead of to 2.52.

6. Conclusions

In this paper the ConRTR algorithm of Groër et al. (2009) is studied. It is showed that some ConRTR solutions that were provided by Groër et al. (2009) are infeasible, because service times are not considered in the travel time of vehicles. I could not conclude that the algorithm can return better solutions by using different parameters than stated in Groër et al. (2009). The random element in the diversification phase has a very large influence on the performance of ConRTR. It is therefore recommended to perform ConRTR multiple times for the same instance.

I have developed an algorithm that decreases the maximum arrival time differential of a ConRTR solution. I showed that the total travel time would increase with 10% when the maximum arrival time differential decreases with 90% to 100%. If the total travel time may not increase more than 5%, the maximum arrival time differential could still decrease with 80% to 90%.

Table 8 **Results decreasing differential**

	Original max. differential	New max. differential	Improvement max. differential	Original total travel time	New total travel time	Increase total travel time
problem1	8.38	0	100%	1,034.64	1,106.10	7%
problem2	17.79	0	100%	1,119.62	1,227.82	10%
problem3	12.76	0	100%	919.70	993.33	8%
problem4	31.8	0	100%	1,746.78	1,899.09	9%
large problem	31.07	2.34	92%	3,693.14	4,020.87	9%
problem1	8.38	1.01	88%	1,034.64	1,086.76	5%
problem2	17.79	3.67	79%	1,119.62	1,175.95	5%
problem3	12.76	1.51	88%	919.70	965.91	5%
problem4	31.8	4.76	85%	1,746.78	1,835.32	5%
large problem	31.07	5.79	81%	3,693.14	3,877.80	5%

References

Groër, C., B. Golden, E. Wasil. 2009. The Consistent Vehicle Routing Problem. *Manufacturing & Service Operations Management* Vol. 11 No. 4 630-643.

Li, F., B. Golden, E. Wasil. 2005. Very large-scale vehicle routing: New test problems, algorithms, and results. *Comput. Oper. Res.* **32** 1165-1179.

Yellow, P.C. 1970. A computational modification to the savings method of vehicle scheduling. *Oper. Res. Quart.* **21** 281-283.

Appendix A

The instances that are used for the experiments with changing parameters are specified in this appendix.

The location and demands for all customers on all days are in the table on the next page.
Location depots for all problems is (0,0).
Service times for all customers, all days and all problems is 2.
Travel times are the Euclidean distances.
The capacity of vehicles is 80.
The maximum travel times of vehicles is 100.

customer	problem 1		problem 2		problem 3		problem 4		demand	demand	demand	demand	demand
	x	y	x	y	x	y	x	y	day 1	day 2	day 3	day 4	day 5
1	-13.62	11.07	-7.42	-4.34	10.00	15.16	25.68	28.74	7	7	7	0	7
2	-11.99	13.84	-2.68	1.74	9.36	14.22	21.81	22.53	30	30	30	30	0
3	-12.02	13.66	13.73	-9.39	7.91	14.39	21.11	24.33	0	16	16	16	16
4	-14.05	14.21	3.14	10.14	6.36	14.85	16.35	23.32	9	9	9	0	9
5	-14.69	12.43	0.83	-2.28	9.82	11.95	25.85	10.28	0	21	21	0	21
6	-14.41	13.19	7.56	-0.16	7.75	15.60	3.48	12.49	15	15	15	0	15
7	-13.32	14.23	-9.37	11.98	5.66	11.68	21.15	29.59	0	19	19	19	19
8	-11.86	12.67	-3.11	-7.82	9.75	11.34	12.03	29.31	23	23	23	0	0
9	-13.11	-12.87	-8.55	9.69	5.74	3.40	22.65	3.91	11	0	11	11	11
10	-13.21	-13.59	12.12	-6.98	4.24	3.50	25.68	5.32	5	5	0	5	0
11	-12.33	-11.09	-10.9	13.50	5.07	2.65	15.26	8.61	19	0	19	19	0
12	-11.78	-11.54	-1.33	-11.7	5.49	0.87	1.06	2.29	29	0	0	0	29
13	-14.13	-11.86	13.96	9.43	5.84	3.42	13.19	15.86	0	23	23	23	0
14	-14.97	-9.59	10.69	14.87	3.81	2.57	2.91	18.63	0	21	21	0	21
15	-13.02	-14.33	-2.25	-13.45	2.86	3.22	2.15	11.32	10	0	10	0	10
16	-13.16	-13.21	-9.79	-14.79	3.83	1.04	14.61	19.75	0	0	15	15	15
17	-4.26	1.85	-1.31	8.35	15.89	2.56	7.56	3.66	3	3	3	3	0
18	-4.96	1.82	-8.15	6.68	13.77	2.20	21.72	2.21	41	41	41	0	0
19	-1.09	-1.49	-5.87	-0.95	13.00	3.60	10.31	23.41	9	9	0	9	9
20	-2.54	1.43	6.11	12.99	14.23	1.48	5.95	28.74	28	0	28	28	0
21	-4.94	-0.16	4.32	7.01	15.62	2.06	15.00	3.54	8	0	8	8	8
22	-3.19	1.82	6.31	0.91	14.37	2.44	26.41	15.22	8	8	8	8	0
23	-3.81	1.61	-2.92	-12.24	14.44	3.68	18.96	16.92	0	16	16	0	16
24	-2.24	1.08	-12.89	-12.34	15.38	3.69	27.64	4.78	10	10	10	10	10
25	-3.50	-2.00	2.88	-3.10	16.55	0.89	26.27	22.39	28	28	28	0	28