A Savings Based Method for Real-Life Vehicle Routing Problems

Alexander Poot
ORTEC Consultants BV, PO BOX 490,
2800 AL Gouda, The Netherlands
and
Econometric Institute, Erasmus University Rotterdam,
PO Box 1738, 3000 DR Rotterdam, The Netherlands

Goos Kant
ORTEC Consultants BV

Albert P.M. Wagelmans
Econometric Institute
and
Rotterdam Institute for Business Economic Studies,
PO Box 1738, 3000 DR Rotterdam, The Netherlands

August 1, 1999

Econometric Institute Report EI 9938/A

Abstract

This paper describes a Savings Based algorithm for the Extended Vehicle Routing Problem. This algorithm is compared with a Sequential Insertion algorithm on real-life data. Besides the traditional quality measures such as total distance traveled and total workload, we compare the routing plans of both algorithms according to non-standard quality measures that help to evaluate the "visual attractiveness" of the plan. Computational results show that, in general, the Savings Based algorithm not only performs better with respect to these non-standard quality measures, but also with respect to the traditional measures.

Keywords: distribution; road transport; vehicle routing

1. *Introduction*

Ever since Dantzig and Ramser [4] first studied the vehicle routing problem (VRP) in 1959, researchers have spent a lot of time and effort on developing methods to tackle this problem. This is due to the fact that the VRP plays a central role in distribution management. In its most simple form, the problem is concerned with the construction of a plan that consists of trips, starting from a central depot, for vehicles servicing customers with known demand. The objective is to minimize total cost without exceeding vehicle capacity.

Fisher [6] describes the history of the development of solution approaches to the basic VRP. The first generation, roughly developed in the 60's and early 70's, consisted of simple heuristics such as savings and insertion methods. In the mid 70's, the second generation started to emerge. Although still approximation methods, these approaches were based on solving one or more related problems to optimality. The third (current) generation has two main streams: artificial intelligence based heuristics and exact optimization.

Many extensions of the basic VRP have been studied in the last decades (see Bodin et al. [2] for a survey). These extensions include complications such as time constraints, multiple capacity constraints, a heterogeneous fleet and multiple depots. Some of the solution approaches for the basic problem can be adapted to deal with additional constraints, but this is typically much easier for relatively simple heuristics, than for more sophisticated approximation and optimization methods. In this sense, the simple heuristics are much more flexible. Flexibility is especially important when the algorithm is going to be applied to many different types of VRP's. Obviously, this is the case for commercial vehicle routing systems that are to be installed at many different clients.

In this paper we describe a method to solve real-life vehicle routing problems, developed at ORTEC Consultants BV, a Dutch consultancy firm specialized in applied operations research. The method has been implemented in SHORTREC Distriplanner® (SDP), a commercial vehicle routing system, which ORTEC has been selling, mainly to European, Asian and American clients, since the mid 80's. Well-known clients include Nedlloyd and BP Amoco. The latter company uses SDP to route its fuel trucks worldwide. Through interaction with its many different clients, ORTEC has obtained expertise on real-life vehicle routing problems, which differs significantly from the expertise of those researchers who mainly study vehicle routing problems in an academic environment and who deal only occasionally (or never) with practical problems. Although we recognize the value of such academic research, we believe that, because of our different point of view, we can make valuable contributions to the literature on vehicle routing.

As mentioned above, many extensions of the basic VRP have been studied in the literature. However, from our experience at ORTEC, we found that real-life problems often involve several non-standard constraints, some of which are never mentioned in the literature, while others are only analyzed isolated, i.e., not in combination with other non-standard constraints. In this paper, we explain how we have adapted the

Savings algorithm, usually contributed to Clarke and Wright [3], to deal with the non-standard constraints. This algorithm has been implemented in recent versions of SDP, where it is available as an alternative to an adaptation of another well-known simple heuristic, the Sequential Insertion algorithm (see Solomon [11]). The reason why the Savings Based method has been added is that the insertion method, SDP's original algorithm, performs satisfactorily overall, but for some clients it produces poor results. It is important to mention that in this context "poor" does not only refer to traditional quality measures such as total costs, total number of vehicles used or total distance traveled, that are usually used to evaluate the quality of a vehicle routing plan. In practice these measures are important, but we have noticed that our clients often use additional measures to decide whether a plan is acceptable or not. In particular, a plan must be "visually attractive". For example, clients tend to dislike plans in which many trips cross each other. They prefer visually attractive plans, because to them such plans are logical and therefore they trust these plans (even if such plans perform worse than other plans with respect to the traditional measures). Since the insertion method sometimes produces visually unattractive plans, we have developed the Savings Based method as an alternative.

This paper contributes to research on VRP on three levels: we have developed a Savings Based algorithm (1) that was compared with a Sequential Insertion Algorithm (2) using non-standard quality measures (3) on real-life data. It turns out that, in general, the Savings Based algorithm not only performs better with respect to the visual attractiveness of the plans but also with respect to the traditional quality measures.

The remainder of this paper is organized as follows. In Section 2 we describe the extended VRP and we will focus on the extensions to the VRP's mentioned in the literature. In Section 3, we describe the non-standard measures to evaluate the visual attractiveness of a plan. In Section 4, we briefly describe the Sequential Insertion algorithm followed by the Savings Based algorithm in Section 5. In Section 6, we describe the data that we use to compare the algorithms. In Section 7 we present the computational results followed by the conclusions in Section 8.

2. Description of the extended VRP

The standard constraints, such as vehicle capacity and maximum trip duration, have received a lot of attention in the literature. Besides these constraints, we deal in practice with several non-standard constraints, some of which are never mentioned in the literature, while others are only analyzed isolated. In this section, we first discuss the typical non-standard constraints, which are explicitly considered in this paper. Secondly, we mention some additional constraints that we also encounter regularly, but less often, in practice. These constraints will not be considered in the remainder of the paper.

2.1 Typical non-standard constraints

The vehicle fleet is assumed to be *fixed* and may be *heterogeneous*. This means that the vehicles may differ with respect to capacity, maximum workload, maximum number of customers per trip, cost per kilometer and fixed cost. Apart from the difference in vehicle capacity, we also deal with *multiple capacity constraints*. So in general, each customer has a demand that is expressed in multiple sizes (for example weight, volume and number of pallets).

Furthermore, there may be *vehicle type constraints*, i.e., some customers can only be served by certain vehicles. Sometimes a customer needs special treatment, which means that only vehicles of a given type can serve this customer. For example, special equipment may be needed to unload the order of the customer. For each customer it is given which vehicle types can serve the customer, and each vehicle has a given vehicle type.

Another constraint between vehicles and customers is the so-called *region constraint*. Some drivers are familiar with a specific customer or region, so it is preferred that these drivers (vehicles) serve the customers in the given region. This information is usually dealt with by assigning certain customers and vehicles a region number. A customer with a region number should preferably be served by a vehicle with the same region number, but it can also be served by a vehicle without a region number. A customer without such a number can be served by any vehicle. A vehicle with a region number can only serve customers with the same region number or customers without a region number.

It is also possible that a group of customers should be *served first (or last) in a trip*. This means that all customers that are served before (after) a customer belonging to this group, are themselves also a member of this group.

There may be *backhaul* customers, i.e., customers who do not have a demand, but a supply of a given good. These supplies are typically much larger than the demands.

Finally, each customer can have one or two *time windows* (service intervals). In the literature, it is very exceptional that customers have more than one time window.

2.2 Other non-standard constraints

Other non-standard constraints that we often encounter in practice are

- limited availability of the good(s) at the depot(s)
- forbidden product combinations (some products can not be transported simultaneously in the same vehicle)
- vehicles compartments (each customer or product has to be assigned to one or more compartments)

These constraints will not be considered in the remainder of this paper. Furthermore, we will restrict ourselves to problems with a single depot in which each vehicle is allowed to leave the depot only once a day. In practice, we sometimes encounter problems with multiple depots and vehicles may be allowed to leave a depot more than once. However, in our current implementation of the Savings Based method this is not possible (our Sequential Insertion algorithm does allow for these features).

3. Non-standard quality measures

As mentioned before, a planner (client) does not only use the traditional quality measures such as total cost, the number of vehicles used, the total distance and the total workload, to evaluate a vehicle routing plan. Therefore, we discuss in this section several non-standard quality measures that will be used, together with the traditional measures, to analyze plans and to compare different algorithms.

3.1 Measures for visual attractiveness

The visual attractiveness of a vehicle routing plan plays an important role in the decision whether or not to accept the plan. As a tool the planner may use a drawing of the trips onto a (virtual) map. For instance, SDP has a graphical user interface that enables the planner to make such a drawing. In this paper, we use several measures to capture the visual attractiveness of a plan, which can be divided into the following four groups:

1. Center of gravity	(*Av. not closest center*)
2. Convex hull	(*Av. in convex hull*)
3. Average distances	(*Av. dist. to center*, *Av. dist. between*)
4. Crossings	(*Tot. cross. between trips*, *Av. cross. within trip*)

The measure *Av. not closest center* gives the average number of customers in a trip that are closer to the center of gravity of another trip than to the center of gravity of the trip itself. Here, the center of gravity of a trip is the center of gravity calculated from the coordinates (in the plane) of the locations of the customers in the trip. Note that the location of the depot does not influence the center of gravity. The idea is that a plan is visually more attractive when *Av. not closest center* is relatively small.

The measure *Av. in convex hull* quantifies the average number of customers per trip that are contained in the convex hull of another trip. Here, the convex hull of a trip is the convex hull formed by the locations of all customers served in the trip. The idea is that it is visually more attractive to have very few customers contained in the convex hull of another trip.

The measures *Av. dist. to center* and *Av. dist. between* use the distance between the locations of the customers to evaluate the plan. The former is the average distance of the customers to the center of gravity of the trip. The latter is the average distance between any two customers in a trip (not necessarily served after each other). The idea is that a planner will prefer vehicle routing plans in which customers served in the same trip are relatively close to each other or to some central point.

To evaluate the number of crossings we have developed two measures, one at the global level and one at the trip level. The first, *Tot. cross. between trips*, is the total number of crossings of different trips, where we may count more than one crossing per pair of trips. The second, *Av. cross. within trip*, is the average number of crossings within a trip, where crossings that occur on the stretch from the depot to the first customer in the trip or on the stretch from the last customer to the depot, are not taken into account. Since we use the coordinates of the locations of the customers to determine the crossings, and not a real network of roads, crossings that we count this way are only an approximation of the actual number of crossings that occur. However, this measure is much easier to calculate (using an algorithm from computational geometry) while the approximation turns out to be reasonable. Both *Tot. cross. between trips* and *Av. cross. within trip* should preferably have low values.

3.2 A measure for region constraint violation

In addition to the measures discussed in the preceding subsection, we propose a measure to evaluate the extent to which the region constraints are violated. Strictly speaking, region constraints can not be violated. However, assigning a customer with a region number to a vehicle without such a number is allowed but not preferred. The total number of such assignments as a percentage of the total number of customers with a region number is a measure for the (weak) region constraint violation. This measure should be corrected for situations where there are no vehicles of a given region number available or when there are too many customers in one region.

4. The Sequential Insertion algorithm

The Sequential Insertion algorithm forms the basis of all insertion algorithms. Solomon [11] describes various variants of this algorithm, while Potvin and Rousseau [9] describe the parallel version. The main idea behind the algorithm is to add non-served customers to the current plan by inserting them at the "best" position. In contrast to the parallel version, the sequential version constructs one trip at the time. Once it is not possible anymore to insert a customer into the current trip, the algorithm starts with a new trip. In this section we briefly describe the version of the Sequential Insertion algorithm as it is implemented in SDP. In this description, terms such as "a largest vehicle" and "smaller vehicles" are used. This refers to the vehicle capacity constraint(s). In case of multiple capacities, some ordering of the capacity types is given and "larger" and "smaller" is defined with respect to a lexicographical ranking of the vehicles based on this ordering.

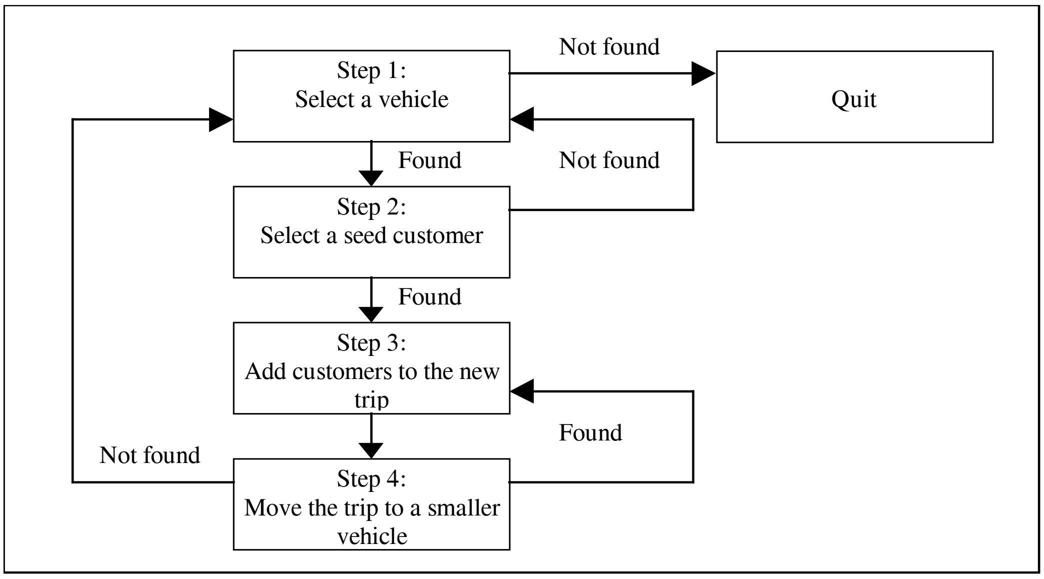

Figure 1: Steps of Sequential Insertion algorithm

SDP's Sequential Insertion algorithm (SI) consists of the following four steps (illustrated in Figure 1):

> *Step 1: Select a vehicle*
>
> We select a largest vehicle that does not perform a trip and which has not been tried before.
>
> *Step 2: Select a first customer for this vehicle (seed customer)*
>
> We select the customer that is most difficult (for example farthest away from the depot or smallest service interval) and which is feasible in the vehicle. If we fail to select a customer we go to Step 1, otherwise assign this customer to the current vehicle.
>
> *Step 3: Add customers to the new trip*
>
> First we build a list of candidates to insert into the trip. A candidate is a non-served customer who is feasible according to vehicle type, capacity and region constraints and that is not located "too far" from the current seed customer. We calculate for each candidate its insertion cost, i.e., the extra distance needed to serve the customer. We sort the items in the list in non-decreasing order of insertion cost. Each time we try to insert the next candidate at its "best" feasible insertion position in the trip. If we can not find a customer to insert for a fixed number of times, then we go to Step 4.
>
> *Step 4: Move the trip to a smaller vehicle*
>
> Here we try to move the trip to a smaller vehicle of the same vehicle type and region number, to save vehicle capacity, without violating any constraint. If we have moved the trip to another vehicle, then we go back to Step 3, else we go back to Step 1.

We emphasize that we leave out a lot of details that are important for the practical performance of the algorithm. For example, we have implemented several fast tests to determine the feasibility of a customer in early stages of the algorithm. The algorithm also allows the user to influence its behavior by giving priority to individual vehicles and customers. It is also possible to change the criteria on which the choice of a seed customer is based, or to change the definition of the insertion costs.

One of the main advantages of the above algorithm is that it is very flexible and robust. For example, it is usually easy to deal with client specific constraints within the general framework.

Although the results that we obtain with the Sequential Insertion algorithm are good in general, we have noticed that in some cases it performs quite poor. In particular, it sometimes produces vehicle routing plans that are visually very unattractive. Therefore, we have experimented with a Parallel Insertion algorithm described in Potvin and Rousseau [9]. It turned out, however, that the improvements over Sequential Insertion were only marginal. Apparently, instead of sticking to the insertion approach, a significantly different method should be used. Therefore, we decided to develop an algorithm based on the old savings idea, which will be described in the next section.

5. The Savings Based algorithm

5.1 Introduction

To overcome the main disadvantage of the Sequential Insertion algorithm we have developed a Savings Based algorithm that constructs trips simultaneously and still generates a plan within a reasonable amount of time.

As mentioned before, the original Savings algorithm itself is usually attributed to Clark and Wright [3]. The main idea behind the algorithm is to start with each customer in a separate trip and then try to find improvements on this solution ("savings") by combining the customers of two trips into one trip without changing the order in which the customers are visited. Contrary to the Sequential Insertion algorithm, the Savings algorithm constructs the trips simultaneously.

The saving can simply be the actual saving in distance or workload. Golden et al. [7] used adjusted savings in case of a heterogeneous vehicle fleet. In contrast to real-life situations, however, they assumed unlimited availability of each vehicle type. Their adjustment consisted of the incorporation of savings in fixed vehicle cost and the incorporation of opportunity savings for unused capacity. Altinkemer and Gavish [1] introduced the Parallel Savings Algorithm. Instead of selecting only the highest saving in each iteration, they solve a maximum weighted matching problem to determine a set of savings with maximum total value. Desrochers and Verhoog [5] use this idea in their Matching Based Savings Algorithm. The difference is that they do not implement all savings corresponding to combinations in the maximum matching, but only the one with the highest value. Besides the original savings, they also used the adjusted savings suggested by Golden et al.

5.2 Adjustments to the original Savings algorithm

The main idea behind our Savings Based algorithm is still to combine two trips with the highest saving, but we have made several adjustments necessary to be able to cope with all constraints mentioned in Section 2.

First of all, we deal with a *fixed* and *heterogeneous* vehicle fleet. So, each time we combine two trips, we have to select a vehicle that is best suitable for the trip. We distinguish between dummy and real vehicles. A real vehicle actually exists, while a dummy vehicle is a copy of a real vehicle that is used in the first step of the algorithm to initialize the trips. Any customer that is still assigned to a dummy vehicle at termination of the algorithm will not be served. During the execution of the algorithm, when two trips are combined, the combined trip is always assigned to a real vehicle. Hence, once a real vehicle serves customers, these customers will remain to be served by a real vehicle until the algorithm terminates. In each iteration, we select the best possible combination of two trips. With a heterogeneous fleet, however, it is often too time consuming to check the feasibility of each combination exactly, because this depends on the vehicle to which the combination is assigned. Therefore, at early stages in each iteration, we perform fast tests with the purpose to eliminate combinations from further consideration. Only at a later stage, we perform exact, more elaborate feasibility tests for relatively few combinations.

Our second adjustment consists of a *preprocessing step* that we have added to reduce the size of the problem instance (the initial number of trips). In this step we partition the set of customers into groups and in each group we try to combine as many customers as possible. Two customers are in the same group if and only if the following conditions hold:

1. their postal codes match
2. their service intervals overlap
3. their region numbers match
4. their feasible vehicle types match

Since the customers have service intervals, we have to *avoid unnecessary waiting time*. Waiting time will normally not decrease in the next iteration of the algorithm (most customers will keep the same predecessor and successor). Therefore, as our third adjustment, we have added a maximum waiting time constraint. We only allow combinations of trips for which the total waiting time of the combined trip is less than a given value.

Fourth, we try to *remove unnecessary waiting time* in the combined trip by applying a local search improvement method.

As a last adjustment, we have *incorporated a region factor and a group bonus* in the function that calculates the savings. In this way the algorithm is more likely to assign customers with a region numbers to a vehicle with the same region number and customers in the same group are more likely to be scheduled after one another.

5.3 Outline of the Savings Based algorithm

We are now going to describe the main steps (illustrated in Figure 2) of the Savings Based algorithm in more detail. In this description the initial schedule is assumed to be empty, but this is not essential. The Savings Based algorithm (SB) consists of the following nine steps:

> <u>*Step1: Creation of dummy vehicles*</u>
> For each customer, create a dummy vehicle and assign the customer to a new trip in this vehicle. A dummy vehicle is always a copy of a real vehicle and there may be multiple copies of the same real vehicle. The choice of which real vehicle is copied will affect only the next step of the algorithm, because that is the only step combined trips are allowed to be assigned to a dummy vehicle, instead of a real one. For each customer, select the vehicle with smallest (but feasible) capacity with the same region number.
>
> <u>*Step 2: Preprocessing: Combinations of customers in the same group*</u>
> Try to decrease the size of the problem by combining trips that contain customers that belong to the same group. However, do not combine trips if the total number of customers exceeds a given upper bound. (If there are several trips with many customers, it may be impossible to find a feasible assignment

to vehicles later on.) A combined trip is assigned to one of the dummy vehicles of the separate trips.

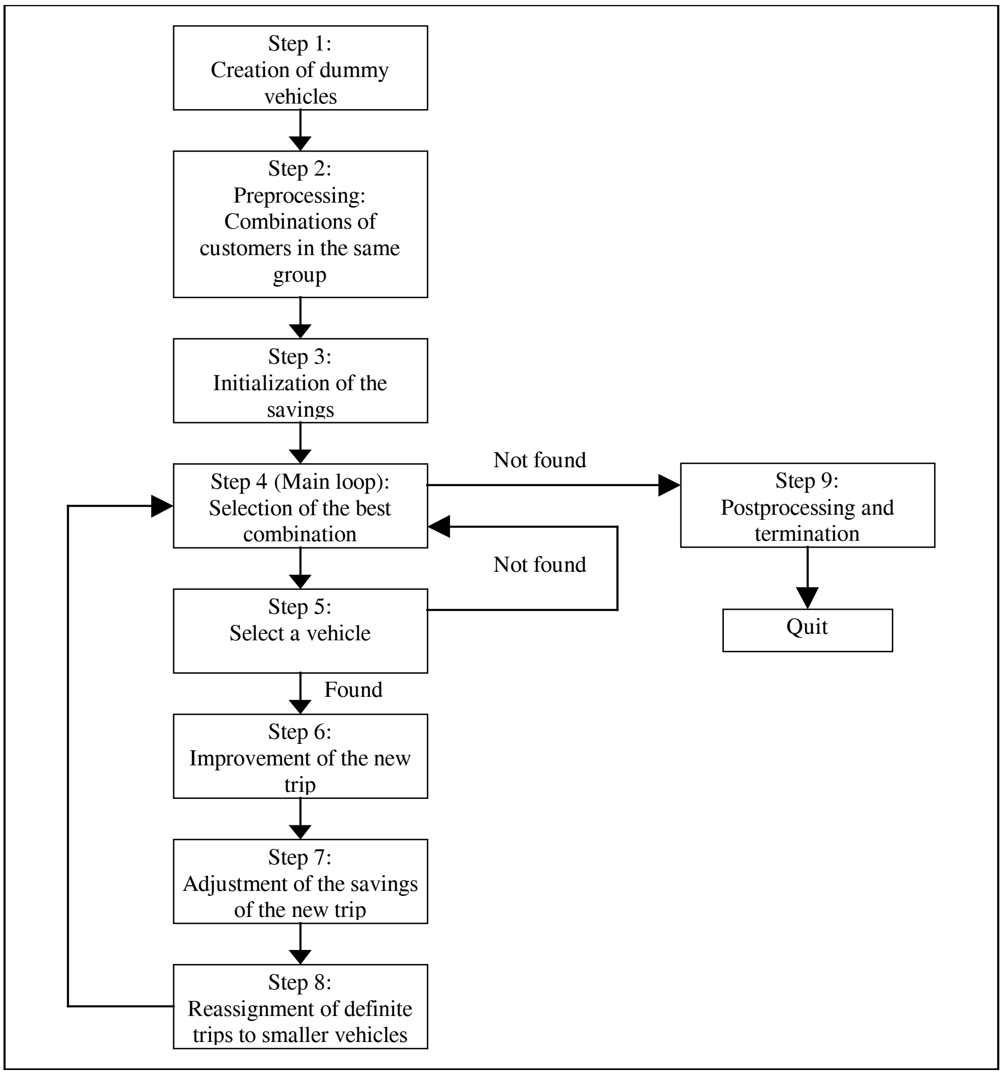
Step 1:
Creation of dummy vehicles
Step 2:
Preprocessing: Combinations of customers in the same group
Step 3:
Initialization of the savings
Step 4 (Main loop): Selection of the best combination
Not found
Step 9: Postprocessing and termination
Not found
Step 5:
Select a vehicle
Quit
Found
Step 6:
Improvement of the new trip
Step 7:
Adjustment of the savings of the new trip
Step 8:
Reassignment of definite trips to smaller vehicles

Figure 2: Steps of Savings Based algorithm

Step 3: Initialization of the savings

For every combination, test if it satisfies all of the following requirements:

1. The maximum load of the combined trip does not exceed the capacity of the largest real vehicle.
2. There exists at least one vehicle that is feasible for all orders with respect to vehicle type constraints.
3. The fixed positions (first or last in the trip) of the orders are not violated.
4. The total time of the combined trip does not exceed the maximum workload of the vehicles.
5. An estimate of the total waiting time does not exceed a given value (to avoid excess waiting time).

If a combination passes the test, we determine the saving using a function that we discuss later in this section. Combinations that fail the test are marked "infeasible". (Note that this fast test is not strict: there may be combinations that are not yet marked "infeasible", but that will turn out to be infeasible later on.)

Step 4: Selection of the best combination (Main loop)
Select the feasible combination with the highest positive saving. Since this combination should be assigned to a real vehicle, it has to be checked whether at least one feasible real vehicle is available (the vehicle is empty or it contains one of the two separate trips). Perform, in a similar manner as in the previous step, a quick test. (As before, the test does not guarantee feasibility, but it is stronger than the test in the previous step, because we can use the characteristics of a specific real vehicle.)

If the test fails for every available real vehicle, mark the combination "infeasible" and repeat this step. If all combinations are infeasible, go to Step 9.

Step 5: Select a vehicle
Select a real vehicle that is feasible with respect to all constraints. If it turns out that there is no feasible real vehicle available mark the combination "infeasible" and go to Step 4.

Step 6: Improvement of the new trip
Try to improve the trip by moving strings (of variable length) of consecutive customers to different positions in the trip.

Step 7: Adjustment of the savings of the new trip
Calculate the savings of all combinations involving the new trip. (Note that it is not necessary to calculate all savings from scratch if the combined trip has the same first or last customer as one of the separate trips.) Also perform the fast feasibility test described in Step 3.

Step 8: Reassignment of definite trips to smaller vehicles
To save vehicle capacity, try to assign trips to a smaller vehicle if it is clear that they can not be combined with any other trip. Go to Step 4.

Step 9: Postprocessing and termination
Remove all customers that are assigned to trips in dummy vehicles. These customers remain unserved. The algorithm terminates.

5.4 Calculation of the savings

In Step 3 and Step 7 we determine the saving for a combination of two trips. Recall that the saving of combining trip *j* after trip *i,* as used in the original savings algorithm of Clark and Wright [3], is defined as follows:

$$CW_{ij} = d_{i0} + d_{0j} - d_{ij} \qquad (1)$$

where d_{ij} is the distance of the last customer of trip *i* to the first customer of trip *j*, d_{i0} is the distance from the last customer of trip *i* to the depot, and d_{0j} is the distance from the depot to the first customer in trip *j*.

We calculate the saving of combining trip *j* after trip *i* with the following formula:

$$SAV_{ij} = region_{ij} * \left(CW(\alpha)_{ij} + group_{ij}\right) \tag{2}$$

where

$$region_{ij} = 1 + \delta(i,j) * \text{REGION\_FACTOR} \tag{3}$$

$$CW(\alpha)_{ij} = d_{i0} + d_{0j} - \alpha * d_{ij} \tag{4}$$

$$group_{ij} = \beta(i,j) * \text{GROUP\_BONUS} \tag{5}$$

and

$\delta(I,j)$	=	0, if some customers in trips *i* and *j* have different region numbers
	=	number of customers with same region number in trip *i* en trip *j* / total number of customers in trip *i* and trip *j*
$\beta(i,j)$	=	1, if the last customer in trip *i* and the first customer of trip *j* have the same group number
	=	0, otherwise
α	=	scaling parameter for the extra distance
REGION_FACTOR	=	extra bonus for customers with the same region number
GROUP_BONUS	=	extra bonus for customers with the same group number

Poot [8] investigated the influence of each parameter. In his experiments he considered four randomly generated basic data sets, each containing 400 customers and an infinite number of identical vehicles. In our experiments we use the standard saving value, i.e., REGION_FACTOR and GROUP_BONUS both equal to 0 and α equal to 1.

6. Description of the real-life data

To perform a fair comparison of the Savings Based algorithm (SB) and the Sequential Insertion algorithm (SI) we use real-life data of four different companies that currently use SDP. Each company has specific characteristics. Table 1 in Appendix A presents an overview of the data.

The first company (A) delivers goods from a depot that is located central to the locations of the customers. Only 7 or 8 % of all customers (around 400-500) supplies goods, but the average supply is much bigger than the average demand. Company A is the only company that has customers who have to be served first or last in a trip (1% of all customers). The service intervals are very wide; on average around 8 hours. On average only 2% of all customers can not be served by all vehicle types. On the other hand, in data set A_1 40% of the customers has a region number, in A_2 25% and in A_3 20%. The number of vehicles varies from 22 to 30, because based on the number of customers for a given day, the company decides which vehicles may be used to service the customers. There is a large variety in capacity of the vehicles, opposite to the maximum workload of the vehicles. In data set A_1 over 66% of the vehicles has a region number, for the other two data sets almost 50% has a region number.

The second company (B) has less customers to serve than company A. The major difference is that we do not have region- or vehicle-type constraints. Each customer has only one service interval with an average length of 8 hours. Company B has a fixed vehicle fleet that is the same for all three data sets.

All customers of the third company (C) demand goods. The major difference between this company and the other three is that both the numbers of vehicles and customers are much less than for the other companies. Similarly to company B we do not have region and vehicle-type constraints. All customers have the same service interval (from 8:00 until 16:00). The vehicle fleet is the same for both data sets, and the vehicles only differ in their capacity.

The last company (D) has much more customers to serve than the other three companies. The average distance from a customer to the depot is very small compared with all other companies. A relatively large part (10%) of these customers supplies goods. The average size (weight in kg and colli) of the supply, in contrast to company A, is less than the average demand of the customers. Although the average length of the service interval is around 7 hours, there are several customers that have a service interval with a length of only 2 hours. For this company the vehicle fleet also differs for the two data sets. The capacity among the vehicles, in contrast to the maximum workload, is very different. The region constraints are very tight: for every region there is only one vehicle available. Besides the region constraints we also have to deal with vehicle-type constraints. There exist four vehicle types and of all customers only a very small percentage can be served by all four types.

Since companies A and D have vehicle-type and region restrictions (in contrast to companies B and C), the general results for these two companies may be different.

7. Computational results on real-life data

The heuristics SI and SB described in the previous sections have been coded in C, and all tests were performed on a 150^+ MHz personal computer with 64Mb internal memory. For each data set we report the results after the construction phase (SI and SB). We also show the results after applying standard improvement methods as described in Savelsbergh [10] (SI + IMP and SB + IMP). These methods try to improve the plan (by means of total distance traveled, total cost etc.) by exchanging or moving strings of customers within and between trips.

7.1 Results according to traditional quality measures

First of all we use the results obtained by the algorithms according to the traditional measures. These results are reported in Table 2 in Appendix B. We give the number of planned customers, the number of trips, the number of vehicles used, the average number of customers per trip, the total cost, the total distance, the total distance per planned customer, the total workload and the computing.

The results show that for companies B and C the number of served customers is more or less the same for algorithms SI and SB. Companies A and D show a different picture. SB uses less vehicles to serve more customers for company A, but for company D the opposite holds. After the improvement phase the gap becomes smaller. Quite remarkable is the difference in the number of trips and vehicles needed for SI and SB for all data sets of company B: SB performs much better. This number has an enormous impact on the total cost.

The differences in the total distance traveled are enormous. If we focus on the results after the construction phase, then we can conclude that the SB performs much better. For all data sets the total distance is much smaller. This is still true when we compute the average distance per served customer. Although the difference has become smaller after the improvement phase, this result still holds, except for data set M_1.

Overall we can conclude for companies B and C that, for the traditional quality measures, the SB algorithm constructs routing plans that serve more customers with less or an equal number of vehicles, but with much less distance traveled and less or comparable total cost.

Unfortunately such strong results do not hold for all data sets of the other two companies (A and D). Since the presence of region constraints and vehicle-type constraints is the major difference between these companies and companies B and C, this is probably the cause of the different results.

7.2 Results according to non-standard quality measures

Secondly we analyze the results of the two algorithms using the non-standard quality measures described in Section 3.1. Table 3 reports the results for these quality measures.

Observation of the quality measure *Av. not closest center* and *Av. in convex hull* tells us that, except for data sets B_2 and B_3, they behave in the same way. The results show that, except for company C, both measures are smaller for the SB algorithm.

If we focus on the quality measures *Av. dist. between customers* and *Av. dist to center*, we, again, can conclude that company C yields different results than the other ones. For almost all data sets, even after the improvement phase, the quality measures are smaller for the SB algorithm.

From these results we may conclude that the SB algorithm constructs plans in which the customers served by the same vehicle are more clustered.

If we compare the plans using the measures *Tot. cross. between trips* and *Av. cross within trip*, it is clear that the SB performs much better than the SI algorithm. Except for company C, the difference in the number of crossings between trips is enormous. Although this number decreases during the improvement phase, the SI algorithm still does not perform as well as the SB algorithm for these quality measures.

8. Concluding remarks

In this paper we have introduced a Savings Based heuristic for real-life vehicle routing problems. The idea is based on the Savings algorithm of Clark and Wright [3], but several adjustments were presented to cope with real-life constraints. We have integrated this heuristic in a commercial vehicle routing system, called SHORTREC Distriplanner® (SDP). Besides the traditional quality measures we have also introduced some non-standard quality measures to evaluate the "visually attractiveness" of a plan.

We have compared the results of this heuristic with the results of the Sequential Insertion heuristic currently used in SDP. Instead of using well-known test sets, we used ten real-life data sets of four different companies. Since this is real-life data we also dealt with non-standard constraints such as vehicle type constraints, a fixed heterogeneous vehicle fleet and region constraints.

Besides the traditional quality measures such as total distance traveled, total number of customers used, we also considered different non-standard quality measures to evaluate the "visually attractiveness" of the plans.

The results presented in this paper show that the SB algorithm outperforms the SI algorithm for almost all data sets, especially if we look at the non-standard quality measures, such as number of crossings between and within trips.

In this paper we have only used the non-standard quality measures to evaluate the "visual attractiveness" of the plans. It is interesting to investigate whether it is possible to incorporate this idea in the optimization function itself together with the traditional measures such as total cost. It is also interesting to investigate whether the general results on the data sets that contain vehicle-type and region restrictions can be improved by introducing a region factor or a vehicle-type factor into the savings function. We also did not test with different functions to determine the savings. Applying the ideas of Golden et al. [7] seems a straightforward extension. Another interesting extension is to adjust the Savings Based algorithm in such a way that it constructs routing plans that contain more than one trip per vehicle.

Appendix A: Description of the real-life data

Table 1: Characteristics of the real-life data sets

Data set	**A_1**	**A_2**	**A_3**	**B_1**	**B_2**	**B_3**	**C_1**	**C_2**	**D_1**	**D_2**
No. customers	412	436	539	279	222	278	187	188	947	1106
Perc. Supply customers (%)	7	7	8	3	2	1	0	0	10	8
Av. Demand (Pallet)	206	337	237	312	413	420	549	1185	43	37
Av. Demand (Pallet)	83	97	71				4	4	0.08	0.09
Av. Demand (Colli)				2	4	3	4	4	345	325
Av. Supply (Kg)	433	976	1693	1893	1325	3079			11	10
Av. Supply (Pallet)	115	182	389						0.3	0.4
Av. Supply (Colli)				14	2	26			89	84
Perc. First in trip (%)	1	1	0							
Perc. Last in trip (%)	0	1	2							
Av. Length service interval (hh:mm)	7:53	8:23	8:22	8:16	8:02	8:10	8:00	8:00	7:11	7:11
St. dev. Length service interval (hh:mm)	1:52	2:30	2:04	2:05	2:23	2:13	0:00	0:00	2:07	2:04
Av. Distance to the depot (km)	58	62	58	98	99	92	106	107	37	37
St. dev. Distance to the depot (km)	31	35	33	70	68	66	58	60	16	15
No. vehicles	22	29	30	29	29	29	17	17	23	28
Av. Capacity (weight in Kg)	10218	9718	10252	10602	10590	10625	20941	20941	4539	6414
Av. Capacity (Pallet)	1827	1969	1863				33	33	5	6
Av. Capacity (Colli)				69012	69011	69011	81	81	22826	22929
Av. Max. workload (hh:mm)	11:42	10:14	9:58	16:14	16:14	16:14	17:00	17:00	12:00	12:00
St. dev. Max. workload (hh:mm)	1:21	1:06	1:24	3:05	3:05	3:05	0:00	0:00	0:00	0:00

Appendix B: Results on real-life data

Table 2: Results according to traditional quality measures

	No. planned customers	No. trips	No. used vehicles	No. customers per trip	Total cost (gld)	Total distance (km)	Distance per customer (km)	Total workload (hh:mm)	Computing time (mm:ss)
data set A_1									
SI	355	22	22	16.1	18085	7438	21	233:44	0:07
SB	365	22	22	16.6	17633	7042	19	232:09	0:33
SI + IMP	362	21	21	17.2	15599	6180	17	206:54	1:34
SB + IMP	371	21	21	17.7	17038	6917	19	226:38	1:27
data set A_2									
SI	383	32	29	12.0	28536	11309	30	289:30	0:07
SB	432	28	28	15.4	25801	8880	21	260:07	0:26
SI + IMP	434	31	29	14.0	25937	10130	23	277:55	1:01
SB + IMP	435	28	27	15.5	23436	8942	21	258:55	2:07
data set A_3									
SI	398	30	30	13.3	22849	10230	26	282:01	0:06
SB	472	30	30	15.7	23036	9373	20	286:36	0:34
SI + IMP	461	30	30	15.4	22685	9683	21	284:58	2:01
SB + IMP	488	30	30	16.3	22580	9255	19	287:31	2:18
data set B_1									
SI	279	26	19	10.7	49089	10312	37	237:34	0:05
SB	277	17	17	16.3	28327	5328	19	155:27	0:23
SI + IMP	279	21	18	13.3	29009	6629	24	173:56	0:37
SB + IMP	279	17	17	16.4	24904	5334	19	154:06	0:45
data set B_2									
SI	222	29	18	7.7	47451	9883	45	221:36	0:06
SB	222	14	14	15.9	21985	4148	19	124:59	0:17
SI + IMP	222	25	17	8.9	25108	5829	26	150:40	0:43
SB + IMP	222	14	14	15.9	20392	4143	19	123:38	0:34
data set B_3									
SI	278	32	20	8.7	48887	9961	36	234:10	0:06
SB	276	16	16	17.3	27873	5217	19	151:29	0:22
SI + IMP	278	29	19	9.6	31773	7303	26	185:05	0:46
SB + IMP	276	16	15	17.3	25531	5217	19	151:02	0:45
data set C_1									
SI	155	24	17	6.5	21354	7541	49	194:31	0:05
SB	160	17	17	9.4	19344	6667	42	177:11	0:12
SI + IMP	157	24	17	6.5	21105	7404	47	193:19	0:21
SB + IMP	169	23	17	7.3	21231	7296	43	196:11	0:38
data set C_2									
SI	154	23	17	6.7	21269	7547	49	193:13	0:04
SB	156	17	17	9.2	19876	7019	45	180:09	0:11
SI + IMP	157	24	17	6.5	21208	7439	47	193:57	0:19
SB + IMP	170	23	17	7.4	21216	7328	43	195:24	0:33
data set D_1									
SI	935	22	22	42.5	54623	5715	6	205:08	0:17
SB	769	23	23	33.4	49252	3800	5	163:23	2:00
SI + IMP	947	22	22	43.0	42109	4123	4	172:43	4:18
SB + IMP	947	22	22	43.0	42508	3889	4	177:09	7:29
data set D_2									
SI	1106	34	26	32.5	63987	6102	6	227:25	0:20
SB	952	28	28	34.0	56524	4195	4	184:34	2:42
SI + IMP	1106	24	23	46.1	45575	4440	4	189:36	5:25
SB + IMP	1082	25	24	43.3	50260	4290	4	190:35	9:53

Table 3: Results according to non-standard quality measures

	Av. not closest center	Av. in convex hull	Av. dist between customers (km)	Av. dist. to center (km)	Tot. cross. between trips	Av. cross. within trip
data set A_1						
SI	6.1	7.9	20	15	67	3.9
SB	5.0	5.9	20	14	25	1.1
SI + IMP	6.4	6.8	20	15	39	1.9
SB + IMP	6.7	6.8	22	17	30	1.5
data set A_2						
SI	5.4	9.3	19	16	87	2.4
SB	3.9	3.8	17	13	41	1.7
SI + IMP	6.7	10.1	23	17	68	1.1
SB + IMP	4.8	6.0	19	15	49	1.4
data set A_3						
SI	4.0	3.4	16	12	39	2.4
SB	5.9	6.4	18	13	25	1.4
SI + IMP	6.5	10.8	21	16	68	1.3
SB + IMP	6.2	5.7	20	14	30	1.6
data set B_1						
SI	6.6	13.1	33	22	185	1.6
SB	5.0	1.0	26	18	3	0.9
SI + IMP	6.5	9.2	35	23	51	1.0
SB + IMP	5.7	1.8	26	18	3	1.2
data set B_2						
SI	5.2	12.3	40	28	186	0.9
SB	5.6	3.2	31	20	6	1.2
SI + IMP	4.0	4.4	33	21	37	0.7
SB + IMP	5.6	3.2	31	20	6	1.1
data set B_3						
SI	5.8	11.2	35	22	183	1.0
SB	6.6	4.8	28	20	10	1.1
SI + IMP	5.1	7.1	31	20	43	0.4
SB + IMP	6.6	4.8	28	20	10	1.1
data set C_1						
SI	2.4	1.6	25	16	25	0.1
SB	4.6	4.2	34	25	22	0.8
SI + IMP	2.5	1.6	25	17	24	0.1
SB + IMP	4.4	3.1	34	24	26	0.6
data set C_2						
SI	2.9	2.0	26	17	16	0.3
SB	3.2	3.3	33	24	26	0.5
SI + IMP	2.9	2.0	27	18	17	0.3
SB + IMP	3.6	2.5	28	20	18	0.4
data set D_1						
SI	22.7	100.1	12	10	727	8.1
SB	12.3	35.6	5	5	205	4.2
SI + IMP	14.8	48.5	8	6	216	6.6
SB + IMP	18.6	57.5	7	5	266	5.1
data set D_2						
SI	16.4	70.1	7	5	573	5.5
SB	13.9	26.9	6	4	145	3.8
SI + IMP	20.3	65.3	7	6	244	5.0
SB + IMP	18.4	41.0	7	5	203	5.0

References

[1] K. Altinkemer and B. Gavish, Parallel savings based heuristics for the delivery problem, *Operations Research* 39, 456-469 (1991).

[2] L. Bodin, B. Golden, A. Assad and M. Ball, Routing and scheduling of vehicles and crews – The state of the art, *Computers and Operations Research* 10, 63-211 (1983).

[3] G. Clarke and J.W. Wright, Scheduling of vehicles from a central depot to a number of delivery points, *Operations Research* 12, 568-581 (1964).

[4] G.B. Dantzig and J.H Ramser, The truck dispatching problem, *Management Science* 6, 80-91 (1959).

[5] M. Desrochers and T.W. Verhoog, A new heuristic for the fleet size and mix vehicle routing problem, *Computers and Operations Research* 18, 263-274 (1991).

[6] M. Fisher, Vehicle routing, in M. Ball, T. Magnanti, C. Monma and G. Nemhauser (eds.), *Handbooks in Operations Research and Management Science 8: Network Routing,* Elsevier Science, Amsterdam, pp. 1-33, 1995.

[7] B. Golden, A. Assad, L. Levy and F. Gheysens, The fleet size mix vehicle routing problem, *Computers and Operations Research* 11, 49-66 (1984).

[8] A.Poot, De Toepassing van Savings in de Praktijk, master thesis (in Dutch), Erasmus University Rotterdam, Rotterdam, 1997.

[9] J.Y. Potvin and J.M. Rousseau, A parallel route building algorithm for the vehicle routing and scheduling problem with time windows, *European Journal of Operational Research* 66, 331-340 (1993).

[10] M.W.P. Savelsbergh, The vehicle routing problem with time windows: minimizing route duration, *ORSA Journal on Computing* 4, 146-154 (1992).

[11] M.M. Solomon, Algorithms for the vehicle routing and scheduling problems with time window constraints, *Operations Research* 35, 254-265 (1987).