ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/221704650

The vehicle routing problem with multiple use of vehicles

Article · January 1990

CITATIONS
132

READS
4,739

1 author:

Bernhard Fleischmann
Universität Augsburg
117 PUBLICATIONS **2,771** CITATIONS

SEE PROFILE

All content following this page was uploaded by Bernhard Fleischmann on 05 June 2018.

The user has requested enhancement of the downloaded file.

The vehicle routing problem with multiple use of the vehicles

Bernhard Fleischmann

Universität Hamburg, Fachbereich Wirtschaftswissenschaften,
Von-Melle-Park 5, D - 2000 Hamburg 13, F.R. Germany
June 1990

© Bernhard Fleischmann, Universität Augsburg, 1990, 2010

Scope and Purpose

In the vehicle routing problem (VRP) a fleet of vehicles at a central depot is to be scheduled so as to service a given set of customers. The usual assumption, that every vehicle can run one tour from and to the depot, is not correct for the daily scheduling of delivery tours in an urban area, where demands are high and travel times short: Then a vehicle has to run several tours a day. While this case is considered in most commercial software packages, there is a lack of relevant theory and methods in the literature. The paper proposes a procedure for this problem and reports on applications.

Abstract

We consider the single-depot VRP with a limited heterogeneous fleet of vehicles, where every vehicle can run several tours within a limited time of use. In addition, customer time window constraints are included. We extend the parallel savings procedure to this case and report on computational experiences with large-scale real-life delivery problems.

1 Introduction

The vehicle routing problem (VRP) consists in designing routes or tours from and to a central depot which service a set of customers with given demands, where every tour is restricted by the vehicle capacity and / or a maximal duration, and the total distance, travel time or cost is to be minimized. One of the most important situations where the VRP occurs in practice, is the delivery of consumer goods for daily varying customer sets and demands, requiring daily routing. However, this problem typically involves additional complications which are not considered in the VRP and the usual solution methods:

(a) There is a given fleet of vehicles of different types with, of course, a limited number of vehicles.

(b) A vehicle that returns from a short tour may be used a second or third time. This is important in urban areas, where the vehicle capacity is the limiting factor and travel times are rather short.

(c) There are **time window** constraints with respect to the arrival time at certain customers.

While the extension (c), the vehicle routing and scheduling problem with time windows (VRSPTW), has been studied intensively in the recent literature [6,8,18,19,20], there is little work on problem (a) and nearly none on problem (b).

The best way to deal with problem (a) is the method of Fisher and Jaikumer [9], the only one that guarantees a feasible solution, if any, for a given fleet by solving a general assignment problem. However, this method does not apply to problem (b) which involves a two-stage assignment of customers to tours and of tours to vehicles. The usual heuristics, in particular the **Savings procedure** (SP) in its basic form (see e.g. [2,17]) assume an unlimited homogeneous fleet. Extensions to fleet restrictions have been considered already in the original paper of Clarke and Wright [5], but rarely in the later literature [7], except for short comments, e.g. in [4,15 p. 193 f].

The lack of theory on problems (a) and (b) is the more surprising, as most commercial software packages cover these problems using the Savings procedure [3,12], but with unpublished details. The most common way, as we know from authors of software, seems to be a two-phase iterative procedure: First, a standard VRP is solved, and in the second phase the resulting tours are assigned to the vehicles. If there is no feasible assignment, the VRP is repeated in total or partially with reduced time and/or capacity limits. This is also proposed in the case report [11]. Another way is the **sequential Savings procedure**, which can construct the tours vehicle by vehicle, but which is known for its poor performance compared to the parallel procedure [16 p. 16ff].

The main subject of this paper is the problem (b), the **vehicle routing problem with multiple use of the vehicles** (VRPMU). It always includes problem (a) and may also include time windows. We propose a method that integrates all restrictions relevant to (a), (b) and (c) into a one-phase parallel SP called **Savings procedure for multiple use of vehicles** (SPMU). For the practical problems considered (see Section 6) with up to 360 customers per day, a SP seems to be most appropriate due to its low computational effort. It works for any definition of the savings function which reflects the objective of the VRP. When daily scheduling a given fleet, however, the most important objective is total distance.

The next section summarizes the necessary notations and terms for the VRP and the basic SP. Section 3 explains a concept for considering a given fleet. Section 4 extends this concept for the multiple use of vehicles and presents the SPMU. Section 5 shows how to include time windows into the SPMU. Section 6 reports on several applications and numerical results.

2 A Framework for the Savings Procedure

The VRP with customers $i = 1, \ldots, n$, the depot ($i = 0$) and vehicles $v = 1, \ldots, m$ is characterized by the following data:

q_i	demand of customer i
Q_v	capacity of vehicle v
d_{ij}	distance between depot or customers i, j
t_{ij}	travel time
s_i	stop time at customer i
$t_{\max}$	maximal duration of a tour.

The capacities and demands may be vectors with several components such as weight, number of pallets, etc. The distances and travel times are supposed symmetric. Note that their definition is ambiguous in a road network with speeds varying from arc to arc, as they cannot be minimal at once. The most realistic definition is to determine t_{ij} as the shortest travel time from i to j and d_{ij} as the length of the corresponding path. In the following, we assume the reader to be familiar with the savings heuristic (see e.g. [2,4;5] and for implementation techniques [13,17]). We only outline the main steps and introduce the notations which are required for the extensions in the next sections.

A tour $T = (i_1, \ldots, i_K)$ is a sequence of customers visited by a vehicle, including the travel from the depot to i_1 and from i_K to the depot. It induces the quantity shipped $q(T)$, the length $d(T)$ and the duration $t(T)$ (travel and stop times). T is feasible if $t(T) \leq t_{\max}$ and, for some vehicle v, $q(T) \leq Q_v$. The **reverse tour**, denoted by T', is feasible if and only if T is feasible. If i is the first or last customer in tour T, $T(i)$ denotes the (eventually reversed) tour starting with i, i.e.

$$T(i) = \begin{cases} T & \text{if } i = 1 \\ T' & \text{if } i = i_K \end{cases}$$

A **shuttle tour** is a tour with a single customer. All shuttle tours are supposed feasible. A **schedule** S is a set of feasible tours $T \in S$, such that every customer is contained in exactly one tour. If $T_1, T_2 \in S$, $T1 = (i_1, \ldots, i_K)$ and $T_2 = (j_1, \ldots, j_L)$,

the **combined tour** is denoted $T1 \times T2 = (i_1, \ldots, i_K, j_l, \ldots, j_L)$.

The SP starts with the schedule S_0 consisting of all shuttle tours. The current schedule S is changed by the following basic operation: Let i and j be the first or last customers of two different tours $T_1, T_2 \in S$. Then T_1 and T_2 are replaced by the combined tour which links i and j,

$$T^* = T_1(i)' \times T_2(j)$$

i.e. S is replaced by $S \cup \{T^*\} \backslash \{T_1, T_2\}$.

This operation yields the **savings** $C(i,j) = d_{0i} + d_{0j} - d_{ij}$ with respect to distance and analogous savings with respect to travel time. The definition of the savings function $C(i,j)$, which determines the order of carrying out the basic operations, is the only part of the procedure, where the overall objective function of the VRP is taken into account. According to different possible objectives (minimal distance, time, number of vehicles, cost) there are different definitions of the savings function proposed in literature [2]. Moreover, perturbations of shape parameters can be used to generate several solutions, from which to choose the best one [13,16,17]. In the following, we admit any appropriate savings function C(i,j). But it should be a function of i, j only and not depend on the current schedule. Otherwise, one has to update the savings during every iteration, so that the main advantage of the SP gets lost.

The general scheme of the SP is then as follows:

Initial step
$S = S_0$; order all pairs (i,j) $(i = 2, \ldots, n;\ j = 1, \ldots, i-1)$ from largest to smallest values $C(i,j)$.

Iteration

(a) Proceed to the next (i,j) such that i and j are the first or last customer in different tours $T_1, T_2 \in S$.

(b) If the tour $T^* = T_1(i)' \times T_2(j)$ is feasible, go to step (c), otherwise to the next iteration.

(c) $S = S \cup \{T^*\} \backslash \{T_1, T_2\}$.

The algorithm terminates in step (a), if all (i,j) have been processed.

A major deficiency of the basic savings algorithm is that it does not take into account the available fleet of vehicles. For this purpose, one has to ensure, in step (b), not only the feasibility of the single tours T, but also the feasibility of the schedule S

as a whole, with respect to the fleet. In the next section, a concept of feasibility is proposed for the sequence of schedules occuring in the savings algorithm.

3 Considering a given fleet of vehicles

In this section, we still assume that every vehicle v can run a single tour. In the next section, the multiple use of the vehicles is allowed for. The notation of feasibility of a schedule S is obvious. S is **feasible** if there exists an **assignment** $v(T) \in V$ of the tours $T \in S$ to the vehicles such that

$$v(T) \neq v(U) \text{ for } T \neq U \text{ and } q(T) \leq Q_{v(T)}. \tag{1}$$

However, this definition is not useful in the SP, because the initial schedule S_0 which requires a separate vehicle for every customer, is infeasible in all realistic situations. Therefore, the feasibility of S cannot be postulated at every iteration of the algorithm. One possible way to overcome this difficulty, proposed in [4,7,15] is to assign only the combined tours, which arise in step (b), to the vehicles. Then, the partial schedule consisting of all tours with at least two customers can be kept feasible during the whole algorithm, but all schedules, incl. the final schedule, may contain unassignable shuttle tours. This procedure has the major disadvantage, that already in early iterations, as soon as the number of combined tours equals the number of vehicles, favourable combinations among shuttle tours are blocked. Retrying blocked combinations at every iteration improves the solution, but increases the effort drastically, compared to the basic SP. And even then, the result may be very poor, because the order of the combinations deviates from the order of decreasing savings.

Another way of relaxing the feasibility restriction for the intermediate schedules, which is already proposed by Clark and Wright [5] is to assume an unlimited number of vehicles **of the smallest size**. We generalize this idea for the VRP and in the next section for the VRPMU. At the beginning, we introduce an appropriate set of **fictitious vehicles**, which render the initial schedule S_0 feasible. The range of capacities of the fictitious vehicies is that of the real capacities $Q_1, \ldots, Q_m$. At any iteration, we impose the feasibility condition on the schedule S, that for any v the number of fictitious vehicles of size Q_v or greater, required for schedule S must not increase. This restriction is very weak at the early iterations, when most combinations improve the feasibility, but strengthens gradually, when the tour sizes approach the vehicle sizes. Once a feasible schedule is reached, i.e. all fictitious vehicles are removed, feasibility is maintained up to the end. Of course, like any savings heuristics, this procedure does not guarantee that feasibility is reached at

all, but it attempts to improve the degree of feasibility at every step.

If the vehicles can be used once only, the feasibility condition can be implemented easily without constructing an explicit assignment $v(T)$ using the following lemma. We order the vehicles according to decreasing capacity:

$$Q_1 \geq Q_2 \geq \ldots, Q_m.$$

(If the capacity is a vector, we assume, that the order is identical for all components; this is a realistic assumption about the vehicle sizes.) Moreover, we group the vehicles into classes of identical size, say

$$V_1 = \{1, \ldots, v_1\},\ V_c = \{v_{c-1} + 1, \ldots, v_c\}\ (c = 2, \ldots, k),$$

i.e. $Q_v = Q_{v_c}$ for all $v \in V_c$. For any tour $T \in S$, the class of the smallest feasible vehicles is

$$c(T) = \max\{c :\ Q_{v_c} \geq q(T)\}, \tag{2}$$

and the number of vehicles of size Q_{v_c} or greater, needed for S, is

$$N_c(S) = |\{T \in S :\ c(T) \geq c\}|\ (c = 1, \ldots, k) \tag{3}$$

Lemma

(a) A schedule S is feasible, if and only if

$$N_c(S) \leq v_c \text{ for } c = 1, \ldots, k. \tag{4}$$

(b) The number of fictitious vehicles of size Q_{v_c} or greater, required for S, is

$$F_c(S) = \max(\max\{N_i(S) - v_i :\ i = 1, \ldots, c\}, 0) \tag{5}$$

Proof

As v_c is the number of available vehicles of size Q_{v_c} or greater, (4) is obviously necessary. If (4) holds, the assignment of the tours T to the vehicles $v(T) = 1, \ldots, m$ in the order of increasing $c(T)$ is feasible; because $q(T) \leq Q_1$ for all $T \in S$ by definition, and if there was a $T_0 \in S$ and $v = v(T_0) \in V_c$ $(c \leq 2)$ such that $q(T_0) > Q_v$, then $c(T) < c$ for the first v tours and hence $N_{c-1}(S) \geq v > v_{c-1}$ which contradicts (4). Statement (b) follows immediately from (a).

□

The numbers $N_c(S)$ need to be determined only for the initial schedule. They can be updated at every change of S in step (c) of the SP as follows:

$$N_c(S) = N_c(S) + \begin{cases} -1 & \text{for } c \geq \max(c(T_1), c(T_2)) \\ 1 & \text{for } c(T^*) \leq c < \min(c(T_1), c(T_2)) \\ 0 & \text{otherwise.} \end{cases} \tag{6}$$

The numbers $F_c(S)$ can be determined from $N_c(S)$ recursively:

$$\begin{aligned} F_1(S) &= \max(N_1(S) - v_1, 0) \\ F_c(S) &= \max(N_c(S) - v_c, F_{c-1}(S)). \end{aligned}$$

Therefore, the feasibility condition is satisfied, if $N_c(S)$ does not increase beyond the present value of $F_c(S)$.

In the whole, the consideration of the given fleet requires the following supplements to the basic SP:

In the initial step:

Determine $c(T)$ for all $T \in S_0, N_c(S_0)$ and $F_c(S_0)$ according to (2), (3) and (6), respectively.

In Step (b) of the iteration:

Determine $c(T^*)$; let $\overline{c} = \min(c(T_1), c(T_2))$.

If $c(T^*) < \overline{c}$ and $N_c(S) + 1 - v_c > F_c(S)$ for some c such that $c(T^*) \leq c < \overline{c}$, reject T^* and go to the next iteration.

In step (c) of the iteration:

Update $N_c(S)$ according to (6) and $F_c(S)$ as follows:

$$F_c(S) = \max(F_c(S) - 1, F_{c-1}(S)) \quad (c = \max(c(T_1), c(T_2)), \ldots, m). \tag{7}$$

4 Multiple use of the vehicles

A vehicle that runs several tours on the same day has to stay at the depot between two tours for a certain time for being reloaded. The time of use of the vehicle, from the beginning of its first tour up to the end of its last tour, is limited, e.g. due to the drivers work time. Thus, the VRPMU involves the additional data:

t_0	minimal time between tours
$u_{\max}$	maximal time of use of a vehicle.

In the VRPMU the definition (1) of the feasibility of a schedule S is replaced as

follows: S is feasible, if there exists an assignment $v(T) \in V$ of the tours $T \in S$ to the vehicles such that

$$\sum_{T:v(T)=v} (t(T) + t_0) \leq u_{\max} + t_0 \quad \text{for } v \in V \text{ and} \tag{8}$$

$$q(T) \leq Q_{v(T)} \quad \text{for } T \in S.$$

The determination of an assignment $v(T)$ which satisfies (8) is a difficult problem. It is a generalized bin packing problem, where the vehicles v are the bins of size $u_{\max} + t_0$ and the tours T are the items to be packed with size $t(T) + t_0$. The items bear the capacity index $c(T)$, defined by (2), and can only be packed into a bin $v \in V_c$ with $c \leq c(T)$. To solve this problem, we modify the First-Fit-Decreasing (FFD) algorithm which is known as the most favourable bin packing heuristic [10]. The decreasing "item size" $t(T)$ is only used as a second order criterion, the first criterion being increasing $c(T)$, because tours requiring bigger vehicles must have priority on these vehicles over smaller tours; moreover, the vehicles are scanned in the order of increasing size. We use the variable u_v = time of use of vehicle v.

Assigning the tours of a given schedule to the vehicles:

$u_v = 0$ $(v \in V)$;
for every $T \in S$ in the order of increasing $c(T)$ and, for equal $c(T)$, decreasing $t(T)$:
    for every $v \in V_c$ $(c = c(T), c(T)-1, \ldots, 1)$:
        if $u_v + t(T) \leq u_{\max}$: $v(T) = v$,
                $u_v = u_v + t(T) + t_0$,
                go to the next T;
    if T has not been assigned: **stop**, no assignment found.

A similar assignment procedure is used by Matthäus [15, p.190f], but only for the combined tours, as explained in section 3.

In order to enforce a feasible assignment for the initial schedule S_0, we again introduce fictitious vehicles and minimize their number using the above assignment heuristic. As soon as a fictitious vehicle becomes idle during the savings procedure, it is withdrawn. The feasibility condition, which is checked before every combination, is simply that the new schedule admits an assignment to the current set of vehicles. Thus, like in the procedure of the previous section, the number of fictitious vehicles never increases, except for the initial step. The fictitious vehicles are numbered $v = m+1, \ldots, m+\tilde{m}$ and augment V and the appropriate classes V_c. In the assignment algorithm, they are scanned **after** the real vehicles of the same class.

As the assignment algorithm is rather time-consuming compared to the operations of the SP, we try to avoid running it before every combination. We first try to update the current assignment by assigning only the new tour T^* either to one of the vehicles $v(T_1)$ or $v(T_2)$, relieved of the tours T_1 and T_2, (Test 1) or to any other vehicle (Test 2). Only if these tests fail, a completely new assignment is constructed using the assignment algorithm (Test 3). This way, the assignment algorithm is performed only at a very small proportion of the savings iterations (cf. Section 6).

In the whole, the SPMU requires the following supplements to the basic SP:

In the initial step:
Determine $c(T)$ for $T \in S_0$ according to (2); $\tilde{m} = 0$. Determine an assignment $v(T)$ for $T \in S_0$ using the assignment algorithm. If it fails to find a vehicle for a tour T, set

$$\begin{aligned} \tilde{m} &= \tilde{m}+1;\ v = m+\tilde{m};\ c = c(T); \\ Q_v &= Q_{v_c};\ V_c = V_c \cup \{v\};\ V = V \cup \{v\}; \\ v(T) &= v;\ u_v = t(T)+t_0 \end{aligned}$$

and continue the assignment algorithm.
In Step (b) of the iteration:
`Test 1:`

`Set` $v'(T) = v(T)$ `for` $T \neq T_1,\ T_2$ `and` $u'_v = u_v$ `for` $v \in V$`;`
$u'_{v(T_i)} = u'_{v(T_i)} - t(T_i) - t_0\ (i = 1,2)$`;`
`for` $i = 1,2$`:`
`if` $c(T_i) \leq c(T^*)$ `and` $u'_{v(T_i)} + t(T^*) \leq u_{\max}$`:`
$v'(T^*) = v(T_i)$`;`
$u'_{v(T_i)} = u'_{v(T_i)} + t(T^*) + t_0$`;`
`go to step (c).`

`Test 2:` Search for a vehicle $v'(T^*)$ for T^*, using one iteration of the assignment algorithm. If it is found, go to Step (c).

`Test 3:` Determine a new assignment $v'(T)$ for $T \in S \backslash \{T_1, T_2\} \cup \{T^*\}$ using the assignment algorithm. If it fails, reject T^* and go to the next savings iteration.

In step (c) of the iteration:

$v(T) = v'(T)\ (T \in S)$`;`
$u_v = u'_v\ (v \in V)$`;`

withdraw all unused fictitious vehicles from V and V_c.

The sorting of the tours T of the current schedule according to increasing $c(T)$ and decreasing $t(T)$, as required by the assignment algorithm in step (b), can be accomplished easily, starting from an initial sorting of the tours $T \in S_0$: At every combination in Step (c), T_1 and T_2 are removed and T^* is inserted into the current order.

The ability of the SPMU to create a feasible solution in case of tight vehicle capacity can be improved by strengthening the feasibility condition. It may happen that a fictitious vehicle, charged with a small tour only, is reloaded with additional tours or customers. This augmented infeasibility can be avoided by controlling not only the number of fictitious vehicles, but also their load, which can be expressed by the number FC of customers supplied by the fictitious vehicles. The additional condition is, that FC must never increase. The implementation of this rule in the SPMU is easy: In the initial step, the starting value of FC is registered; in the Tests 1 and 2, the tour T^* can be assigned to a fictitious vehicle only if both $v(T_1)$ and $v(T_2)$ are fictitious. In Test 3, the corresponding number FC' is registered for the new assignment $v'(T)$; a tour is assigned to a fictitious vehicle only, if the number of its customers does not exceed $FC - FC'$. In step (c), FC is updated.

5 Time windows

Time window constraints cause additional difficulties for scheduling the repeated use of vehicles. However, the SPMU can be well extended to this case. We explain the main ideas of the algorithm, which has been used for some of the results in Section 6, but suppress straightforward details of the implementation. For simplicity, we consider only a single time window per customer; the extension to several time windows is easy. We then need the following additional data:

a_i, b_i	time window of customer i (earliest and latest arrival time at the customer),
A_v, B_v	time window of vehicle v (earliest start of the first tour, latest end of the last tour),
$w_{\max}$	maximal waiting time per tour.

The time window constraints restrict the feasibility of both the single tours and the whole schedule. The first aspect, the check on the feasibility of the combined tour T^* in SP, has been dealt with frequently in literature, see e.g. [6,8,18] for the parallel and [20] for the sequential SP. Similarly, we record the following variables for all tours $T \in S$ and the reverse tours T':

$a(T),\ b(T)$	time window of tour T (earliest and latest arrival at the first customer)
$w(T)$	waiting time of tour T
$k(T)$	'kernel time' (duration of the tour without travel time from and to the depot and without waiting time).

T' is feasible w.r.t. the time windows if $a(T) \leq b(T)$ and $w(T) \leq w_{\max}$. The initialization for a shuttle tour $T = (i)$ is

$$a(T) = a_i,\ b(T) = b_i, w(T) = 0,\ k(T) = s_i,$$

the update for the combined tour $T^* = U_1 \times U_2$, where $U_1 = T_1(i)'$, $U_2 = T_2(j)$ is

$$\begin{aligned}
w(T^*) &= \max\{a(U_2) - b(U_1) - k(T_1) - t_{ij}, w(U_1)\} + w(U_2) \\
b(T^*) &= \min\{b(U_1), b(U_2) - k(T_1) - t_{ij} - w(U_1)\} \\
a(T^*) &= \begin{cases} \max\{a(U_1), a(U_2) - k(T_1) - t_{ij}\} & \text{if } w(T^*) = 0 \\ b(T^*) & \text{otherwise} \end{cases} \\
k(T^*) &= k(T_1) + k(T_2) + t_{ij}
\end{aligned}$$

and T^* is feasible, if $T_1(i)'$ and $T_2(j)$ are feasible and

$$\begin{aligned}
w(T^*) &\leq w_{\max} \text{ and} \\
a(U_1) &\leq b(U_2) - k(T_1) - t_{ij} - w(U_1).
\end{aligned}$$

The feasibility of the reverse tour $T^{*\prime}$ is checked analogously. The combination is performed, if at least one of the tours T^* and $T^{*\prime}$ is feasible.

The feasibility of a schedule S now involves not only an assignment of the tours to the vehicles, but also a **feasible sequence** of all tours assigned to the same vehicle. S is feasible if, for $T \in S$, there exists an assignment $v(T)$ satisfying (8) and starting times

$$ST(T) \text{ starting time of tour } T \text{ at the depot}$$

such that

$$a(T) \leq ST(T) + t_{0i} \leq b(T), \tag{9}$$

where i is the first customer of T, and for any $v \in V$ and $\{T_1, \ldots, T_z\} = \{T :\ v(T) = v\}$

$$
\begin{aligned}
A_v &\leq ST(T_1) && (10)\\
ST(T_p) + t(T_p) + t_0 &\leq ST(T_{p+1}) \ (p = 1, \ldots, z-1) && (11)\\
ST(T_z) + t(T_z) &\leq B_v && (12)\\
ST(T_z) + t(T_z) &\leq ST(T_1) + u_{\max}. && (13)
\end{aligned}
$$

Note that the starting times $ST(T)$ are related to the assignment $v(T)$ and change with it, whereas the time window $a(T)$, $b(T)$ and the waiting time $w(T)$ are fixed properties of the tour T. The starting time $ST(T)$ has to be determined in the assignment algorithm simultaneously with $v(T)$. An easy way to do that would be to **append** every tour as the last tour to those tours already assigned to the same vehicle. But this works only, if the tours are considered in the order of their time windows, which may be quite different from the FFD-order, and gives poor results, to our experience. It is preferable to use the order of the tours as in the assignment algorithm in Section 3 and to **insert** the current tour at an appropriate position, as follows:

Determination of the starting time ST(T) in the assignment algorithm:

Let T be the tour and v the vehicle under consideration and i the first customer of T.

If $A_v + t_{0i} > b(T)$ or $B_v + t_{0i} - t(T) < a(T)$, reject v. If $u_v = 0$, set

$$ST(T) = \min(A_v, a(T) - t_{i0}) \tag{14}$$

otherwise, a sequence of tours $T_1, \ldots, T_z$ is already assigned to v. Try to insert T at any position $p = z, z-1, \ldots, 0$, i.e. after T_p or as the first tour on v, as follows: Set

$$ST(T) = \min(ST(T_p) + t(T_p) + t_0,\ a(T) - t_{i0}),\ \text{if } p > 0$$

and $ST(T)$ according to (14), if $p = 0$. Shift the tours $T_{p+1}, \ldots, T_z$ to the right as far as (11) requires; if this is prevented by (9) for some T_q, try the same with T'_q, if feasible. If only (13) is violated, shift the tours $T_1, \ldots, T_p$ to the right as far as possible. If (9) - (13) are satisfied, perform the assignment $v(T) = v$ and go to the next tour, otherwise try the next position p. If all positions have been checked without success, try the same with T', if feasible. If no feasible start times are found, reject v and go to the next vehicle.

Note that the assignment algorithm is used in Test 3 to produce preliminary starting times $ST'(T)$ only, like the assignment $v'(T)$. Only if a complete assignment has been established successfully, $ST'(T)$ and $v'(T)$ replace the current $ST(T)$ and $v(T)$. The above procedure is also used for determining $ST(T^*)$ in Test 2. In Test 1, $T^{*\prime}$ is inserted on vehicle $v(T_i)$ $(i = 1, 2)$ at the position of the (withdrawn) tour T_i only.

Finally, we improve the sequence of customers within all tours of the final schedule using the OROPT procedure as described in [19]. The criterion is distance, but we impose the additional restrictions on the tour T to be improved that (i) the travel time must not increase and (ii) $a(T), b(T)$ remain within the limits given by a_v and the preceeding tours, b_v and the following tours, respectively, the latter being shifted to the right, if necessary. The tours within a vehicle are processed from the last one to the first one.

6 Application and Results

In this section we report on numerical experiences with the SPMU and on some applications. Unfortunately, there are no standard test data for this case in the literature, so that comparative studies are not possible. However, the data for the three problem classes presented in the following are available from the author. All data stem from real-life VRSP problems for daily deliveries. The first case ("Berlin") concerns a carrier delivering miscellaneous consumer goods in West-Berlin. These data from 1985 have also been used in the fleet capacities studies [1,14]. We use them to illustrate the difference between the standard SP and the SPMU. The two other cases ("Duisburg" and "Dortmund") are current applications of our procedure, concerning the delivery of food from a depot in Duisburg/Rhein to supermarkets and of beverages from a depot near Dortmund to retailers.

Table 1 summarizes the characteristics of the three cases. In all cases the multiple use of the vehicles is very important, the average number of tours per vehicle being at least 1.6, a typical situation for deliveries in an urban area such as Berlin or the Rhein-Ruhr area. In the Duisburg case, this ratio is even nearly 3, as most tours are very short (107 minutes in the average); this is due to a small number of customers per tour and to the use of containers allowing for short unloading times.

Time windows occur only in the Duisburg and Dortmund cases. In the former, about two thirds of the customers receive fresh products which are to be delivered by 9.00 a.m. within a time window of 2 or 3 hours. In the latter, nearly all customers have a time window with an average length of 4.5 hours or two time windows, one in the morning and one in the afternoon. The **tightness** of the time windows, i.e. the ratio of the average time window length per customer over the total time horizon,

as defined in [6,20], is about 50% in both cases. Note that the time horizon (13.5 hours) is longer than the maximal time of use of a vehicle (10 hours). These time window structures are quite different from those generated at random [18,20]. We found that they can be well satisfied without waiting times, so we set $w_{\max} = 0$. Then there is no necessity for considering the time windows in the Savings function, as proposed e.g. in [6].

The specified utilizations with respect to capacity and time are:

$$U_{\text{capacity}} = \frac{\text{total weight delivered}}{\sum_v Q_v \cdot (\text{no. of tours of vehicle } v)}$$

and

$$U_{\text{time}} = \frac{\text{total time of use}}{u_{\max} \cdot (\text{no. of vehicles used})}$$

The number of vehicles used is in the average somewhat lower than the number of vehicles available. In fact, in the Dortmund case, one vehicle has been saved permanently by the use of the VRP system. As an effect of the multiple use of the vehicles, the utilization of the used vehicles is well balanced between capacity and time. Only in the Duisburg case, the time utilization is low, as there is a bottleneck of capacity for the delivery of the fresh products in the early morning and excess capacity for the rest of the day. However, the all-day schedule saves about 8% of the total distance, 4% of the time of use and 4% of the tours as compared to the separate planning of the fresh and non-fresh products performed so far by the company. The high utilization of the used vehicles is an indicator of a good quality of the schedule for a given fleet. Note that the values in Table 1 are averages over several days, the utilization at certain days is still higher.

The OROPT procedure yielded a reduction or 2.4% in distance and 1.1% in travel time in the average in the Berlin case. In the other case no remarkable improvements occurred because of the low number of customers per tour.

Table 2 compares the results of the SPMU with the standard parallel SP. The latter was used iteratively to consider the multiple use of the vehicles, in two different ways: First, we used a commercial VRP package which assigns the resulting tours to the existing vehicles, one by one as far as possible, but allows for easy interactive correction of the schedule, in particular switching customers between tours and fixing or dissolving tours for a further iteration. This work, which was carried out by a well-skilled student, took about half an hour for the final schedule of one day; it is documented in [1]. As this procedure is difficult to follow, we automated it in the following way: A schedule for the single use of the vehicles is determined as in section 3, the tours are assigned to the vehicles using the assignment algorithm of

section 4, the tours in the fictitious vehicles are dissolved and rescheduled in the next iteration where $t_{\max}$ is reduced to the maximum free time of any (real) vehicle. The results are worse in any aspect than those of the interactive procedure, indicating that the latter has been well performed. However, the extended savings procedure still reduces the total distance by about 7%, with nearly no change in the other criteria. The reason is, that it selects the best savings that still admit a feasible schedule, as opposed to the iterative fixing and dissolving of tours.

As to the savings function, we implemented the parametric function of Paessens [17] for both distances and travel times. However, all results shown in this paper are based on the original function with distances. We found that parameter variations yield only slight improvements, if any, and are not worth the additional effort. The time-based savings function may be advantageous in case of tight time constraints, but as Table 3 shows, it effected only a small reduction of travel time in all problems considered and of the number of tours in the Berlin case; on the other hand the total distance, which is the main cost factor for a given fleet, increases remarkably, and in the Duisburg and Dortmund Case also the number of vehicles used.

Table 4 shows the computational effort (without OROPT) for selected problems needed by the implementation in MS-Fortran on a Siemens PC with a 386/387 16 MHz processor. The computation times not only depend on the size of the problem but also on the occurence of time windows and on the number of times a new asignment of tours to vehicles is constructed; this equals to the number of successful Tests 3 plus the number of rejected combinations due to unassignable tours. The number of rejections also indicates the difficulty of the problem with respect to the vehicle assignment and the importance of the SPMU compared to the SP. In this sense, the Berlin and the Dortmund case are rather difficult, and the Duisburg case is easy because its short tours admit many assignments.

Conclusions

We have developed an algorithm for the VRPMU which is based on the SP. It is distinguished by the integration of all necessary feasibility checks and the assignment of tours to vehicles into a parallel procedure, which outperforms a multi-phase or sequential SP. It yields satisfactory results for large-scale practical delivery problems at low computation time. It also works well in case of time window constraints, at least for the typical clustered time windows of delivery problems, although some authors report on poor performance of the SP for the VRSPTW with randomly generated time windows [18]. However, a definitive evaluation of the algorithm requires further computational studies.

7 REFERENCES

1. J. Binsfeld, Erprobung eines Verfahrens zur Bestimmung des optimalen Fuhrparks anhand eines Praxis-Falles. Diplomarbeit, Institut für Unternehmensforschung, Universität Hamburg (1989).

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, No.2, pp. 63-201 (1983).

3. M.A.G. Bocxe and C.B. Tilanus, Testing vehicle scheduling programs for milk collection (case study). European Journal of Operational Research 20, pp. 25-33 (1985).

4. N. Christofides, A. Mingozzi and P. Toth, The vehicle routing problem. In Combinatorial Optimizations, N. Christofides, R. Mingozzi, P. Toth and C. Sandi (eds.), pp. 315-338. John Wiley & Sons, New York (1979).

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

6. M. Desrochers, J.K. Lenstra, M.W.P. Savelsbergh and F. Soumis, Vehicle routing with time windows: optimization and approximation. In: Vehicle routing: methods and studies, B.L. Golden and A.A. Assad (eds.), pp. 65-84. North-Holland, Amsterdam (1988).

7. G. Diruf, Ein heuristisches Verfahren zur Berücksichtigung heterogener Fahrzeugeinsatzcharakteristiken in der computergestützten Tourenplanung.
In: Operations Research Proceedings 1983, H. Steckhan et al. (eds.), pp. 241-248. Springer-Verlag, Berlin - Heidelberg (1984).

8. G. Diruf, Kundenzeitschranken in der computergestützten Tourenplanung. Zeitschrift für Operations Research. 24, B207-B220 (1980).

9. M.L. Fisher and R. Jaikumar, A generalized assignment heuristic for vehicle routing. Networks 11, pp. 109-124 (1981).

10. M.R. Garey and D.S. Johnson, Approximation algorithms for bin packing problems: a survey. In: Analysis and design of algorithms in combinatorial optimization, G. Ausiello and M. Lucertini (eds.), pp. 147-165. Springer-Verlag, Wien New York (1981).

11. B. Golden, A. Assad and R. Dahl, Analysis of a large-scale vehicle routing problem with an inventory component. Working paper MS/S 84-015, University of Maryland, College Park (1984).

12. B. Golden, L. Bodin and T. Goodwin, Microcomputer-based vehicle routing and scheduling software. Computers and Operations Research 13, pp. 277-285 (1986).

13. B. Golden, T. Magnanti and H. Nguyen, Implementing vehicle routing algorithms. Networks 7, pp. 113-148 (1977).

14. H. Koriath, Zur Optimierung von Fuhrparkstrukturen bei Berliner Unternehmen des gewerblichen Straßengüter-Nahverkehrs. Doctoral thesis, Fachbereich Informatik, Technische Universitat Berlin (1988).

15. F. Matthäus, Tourenplanung. Toeche-Mittler Verlag, Darmstadt (1978).

16. H. Paessens, Tourenplanung bei der regionalen Hausmullentsorgung. Schriften des Instituts für Siedlungswasserwirtschaft Nr. 26, Universitat Karlsruhe (1981).

17. H. Paessens, The savings algorithm for the vehicle routing problem. European Journal of Operational Research 34, pp. 336-344 (1988).

18. M.M. Solomon, Algorithms for the vehicle routing and scheduling problem with time window constraints. Operations Research 35, pp. 254-265 (1987).

19. M.M. Solomon, E.K. Baker and J.R. Schaffer, Vehicle routing and scheduling problems with time window constraints: efficient implementations of solution improvement procedures. In: Vehicle routing: methods and studies. B.L. Golden and A.A. Assad (eds.), pp. 85-105. North-Holland, Amsterdam (1988).

20. H.R.G. van Landeghem, A bi-criteria heuristic for the vehicle routing problem with time windows. European Journal of Operational Research 36, pp. 217-226 (1988).

A Table 1. Characteristics of the three cases

Case	Berlin	Duisburg	Dortmund
Problems	20	10	5
Customers			
average per day	241	314	95
maximum per day	303	361	109
average per tour	19	2.8	3.8
maximum per tour	52	9	11
Time Windows			
% customers	0	64	97
% tightness	0	46	50
Fleet			
vehicles	9	42	18
vehicle classes	6	3	14
min.-max. capacity	1030-9020 kg	27-45 Contain.	2940-22500 kg
Tours			
average per day	13	114	24
maximum per day	19	131	29
average per vehicle	1.8	2.9	1.6
maximum per vehicle	4	5	3
Utilization			
% of capacity	88.3	94.4	83.5
% of time	92.1	64.3	77.8
% of vehicles	84.4	91.9	81.5

B Table 2. Comparison of different procedures for the Berlin case (All values are averages per day from 20 problems)

	SF + iterative adaptation		SPMU
	automatic	interactive	
total distance in km	694	668	624
total duration in minutes	3404	3262	3253
tours	14.6	12.6	12.8
vehicles used	7.6	7.1	7.1

C Table 3. Impact of the Savings function based on travel times instead of distances (All values are changes in %)

Case	Berlin	Duisburg	Dortmund
total distance	+ 2.5	+ 2.9	+ 1.0
total duration	- 0.6	- 1.0	- 1.0
tours	- 1.9	+ 2.7	+ 1.8
vehicles used	0.0	+ 1.3	+ 5.6

D Table 4. Computation time and effort for vehicle assignment

Case	Berlin				Duisburg			Dortmund	
Time windows	no				yes			yes	
Customers	194	240	270	303	290	344	361	68	109
Computation time (seconds)	12	26	26	32	74	115	63	7	12
No. of checks on vehicle assignment	276	835	417	345	173	243	230	198	302
- Test 1 successful	129	144	153	172	19	42	66	32	32
- Test 2 successful	36	42	59	55	74	92	152	5	23
- Test 3 successful	18	38	47	65	74	96	12	14	29
- rejections	93	611	158	53	6	13	0	147	218

View publication stats