European Journal of Operational Research 66 (1993) 331–340
North-Holland

Theory and Methodology

A parallel route building algorithm for the vehicle routing and scheduling problem with time windows

Jean-Yves Potvin and Jean-Marc Rousseau

Centre de Recherche sur les Transports, Université de Montréal, C.P. 6128, Succ. "A", Montréal, Que., Canada H3C 3J7

Received September 1990; revised May 1991

Abstract: This paper describes an insertion algorithm for the Vehicle Routing and Scheduling Problem with Time Windows (VRSPTW). This algorithm builds routes in parallel and uses a generalized regret measure over all unrouted customers in order to select the next candidate for insertion. Numerical results on the standard set of problems of Solomon are reported as well as comparisons with his sequential algorithm (Solomon, 1987).

Keywords: Transportation; Vehicle routing and scheduling; Time windows; Heuristics

1. Introduction

This paper introduces a new heuristic algorithm for the Vehicle Routing and scheduling Problem with Time Windows (VRSPTW). The VRSPTW has recently been the focus of very intensive research and has been used to model many realistic applications, as described in Bodin et al. (1983) and Solomon and Desrosiers (1988).

The overall objective is to service a set of customers at minimum cost with a fleet of vehicles of finite capacity housed at a central depot. Each customer has a known demand to be services (either for pick-up or delivery but not both) and a time window or time interval for that service. A time window is also included at the depot in order to define a 'scheduling horizon'. Each route must start and end within the bounds of that horizon which is thus acting like a capacity constraint (e.g. by reducing the scheduling horizon, only a few customers can be serviced by the same vehicle).

Since we are dealing with the hard time window case, a vehicle must also wait if it arrives too early at a customer location. Accordingly, the total routing and scheduling costs include not only the total travel distance (or total travel time), but also the waiting time. Hence, spatial and temporal aspects of the problem are both very important in the search for a good solution.

Interesting results have been reported with respect to route building and route improvement

Correspondence to: Jean-Yves Potvin and Jean-Marc Rousseau, Centre de Recherche sur les Transports, Université de Montréal, C.P. 6128, Succ. "A", Montréal, Que., Canada H3C 3J7.

heuristics. For route building procedures, we refer to the interesting surveys of Solomon and Desrosiers (1988) and Desrochers et al. (1988). Description of local search heuristics for improving routes with time windows may be found in Or (1976), Solomon et al. (1988), Potvin et al. (1990), Savelsbergh (1990).

Important recent developments relating to optimal algorithms for the VRSPTW are described in Desrochers et al. (1990). The current state of the technology is such that a few problems with 100 customers have been solved optimally by Desrochers et al. (1990) on a powerful SPARC workstation using their well-known column generation technique. Due to its exponential nature, however, this approach cannot currently solve large problems. In fact, even for problems with 100 customers, it is unlikely that an optimum solution can be found because, for such large problems, the branching and bounding procedure needs to be very efficient. Accordingly, the procedure must be tuned for the specific problem (or class of problems) at hand, which is not an easy task.

Heuristic problem-solving approaches, on the other hand, can solve much larger problems than optimal algorithms (in much less computation time), and research in this area is thus still justified. Of particular interest is a recent paper by Solomon describing a variety of route construction heuristics for the VRPSTW (Solomon, 1987). The results of his study shows that a sequential time–space based insertion algorithm, which is a generalization of the Mole and Jameson's approach to the VRP (1976), has outperformed all other approaches on a set of routing test problems proposed initially by Christofides et al. (1979). Several authors, like Sorensen (1986), have confirmed the success of this approach. The aim of this paper is thus to show how the sequential insertion algorithm of Solomon can be substantially improved, for some problem instances, by using a parallel route building philosophy.

In the following, Section 2 first briefly introduces Solomon's approach to the VRSPTW problem. Section 3 then describes our parallel approach to the problem and introduces a new metric for insertion, which is based on a generalized regret measure over all unrouted customers. Finally, computational results are presented in Section 4 and concluding remarks are made.

2. A sequential approach to the VRSPTW

In this section, we briefly introduce Solomon's sequential approach to the VRSPTW (Solomon 1987). Here, routes are created one at a time. A route is first initialized with a 'seed' customer and the remaining unrouted customers are added to this route until it is full with respect to the scheduling horizon and/or capacity constraint. If unrouted customers remain, the initialization and insertion procedures are then repeated until all customers are serviced.

After initializing the current route with a seed customer, the method uses two criteria $c_2(i, u, j)$ and $c_1(i, u, j)$ to respectively select and insert customer u between adjacent customers i and j in the current partial route. Let $(i_0, i_1, i_2, \ldots, i_m)$ be the current route with i_0 and i_m representing the depot. For each unrouted customer, we first compute its best feasible insertion place in the route as

$$c_1(i(u), u, j(u)) = \underset{p=1,\ldots,m}{\text{optimum}} \left[c_1(i_{p-1}, u, i_p) \right].$$

Next, the best customer u^* to be inserted in the route is selected as the one for which

$$c_2(i(u^*), u^*, j(u^*)) = \underset{u}{\text{optimum}} \left[c_2(i(u), u, j(u)) \right],$$

u unrouted and feasible.

Client u^* is then inserted in the route between $i(u^*)$ and $j(u^*)$. When no more customer with feasible insertions can be found, the method starts a new route, unless it has already routed all customers.

In his paper, Solomon describes three different heuristics based on this insertion approach. The best heuristic minimizes a weighted sum of detour (in time units) and delay to identify the best insertion place for each customer. The selection of the customer to be inserted is then based on a generalization of the savings measure of Clarke and Wright (1964).

Since Solomon uses two distinct initialization criteria for selecting a new seed customer (i.e. farthest customer from the depot or customer with earliest deadline) and four different parameter settings for tuning its insertion and customer selection formulas, the heuristic is applied eight

times to each problem and the best overall solution is selected as the final solution.

Experiments that we ran with Solomon's sequential algorithm demonstrate that, in some cases, the last unrouted customers tend to be widely disseminated over the geographic area. When this happens, the last routes are usually of poor quality. This is a well known intrinsic weakness of myopic sequential approaches. In order to alleviate this problem, we propose instead a parallel approach for initializing and building the set of routes. This new algorithm is described in the next section.

3. A parallel approach to the VRSPTW

The new parallel algorithm makes use of the generic insertion framework of Solomon, but introduces a new customer selection cost as well as a parallel route building philosophy. Here a set of n_r routes is initialized at once, and the remaining customers are then inserted into any one of those routes.

One difficulty with the parallel approach, however, is to determine what the initial number of routes n_r should be. In order to do that, we estimate n_r by applying the sequential algorithm of Solomon only once, using the farthest customer from the depot for initialization and a single parameter setting for tuning the insertion and customer selection formulas (i.e. $(\beta, \alpha_1, \alpha_2) = (1, 1, 0)$, see Solomon (1987) for details). In this way, we obtain an initial estimate for n_r and we can then build n_r routes in parallel. As described below, the set of routes generated by Solomon's algorithm is also used for selecting appropriate seed customers.

The parallel algorithm can now be described with respect to the insertion approach of Solomon, as introduced in Section 2:

1. Initialization is done by selecting the farthest customer from the depot in each one of the routes generated by the sequential algorithm of Solomon (with only one initialization criterion and a single parameter setting). Applying this sequential algorithm only once is thus useful both for estimating the initial number of routes n_r and for identifying appropriate seed customers. By using Solomon's routes, we get a good initial distribution of the n_r seed customers over the geographic area and the insertion of the remaining customers is less likely to leave the last unrouted customers widely apart.

2. For each unrouted customer u, we first compute its best feasible insertion place in each one of the n_r initial routes:

$$c_{1r}^*(i_r(u), u, j_r(u)) = \min_{p=1,\ldots,m} \left[c_{1r}(i_{r_{p-1}}, u, i_{r_p})\right], \quad r = 1, \ldots, n_r;$$

$$c_{1r}(i_r, u, j_r) = \alpha_1 \cdot c_{11r}(i_r, u, j_r) + \alpha_2 \cdot c_{12r}(i_r, u, j_r),$$

$$\alpha_1 + \alpha_2 = 1, \ \alpha_1 \geq 0, \ \alpha_2 \geq 0.$$

Here

$c_{11r}(i_r, u, j_r) = d_{i_r,u} + d_{u,j_r} - d_{i_r,j_r}$,
$d_{k,l}$ = distance (in time units) between k and l,
$c_{12r}(i_r, u, j_r) = b_{u,j_r} - b_{j_r}$,
b_k = current service time at k,
$b_{k,l}$ = new service time at l, given that k is now in the route.

Then, we select u^* such that

$$c_2(u^*) = \max_u \left[c_2(u)\right],$$

$$c_2(u) = \sum_{r \neq r'} \left[c_{1r}^*(i_r(u), u, j_r(u)) - c_{1r'}^*(i_{r'}(u), u, j_{r'}(u))\right],$$

where

$$c_{1r'}^*(i_{r'}(u), u, j_{r'}(u)) = \min_{r=1,\ldots,n_r} \left[c_{1r}^*(i_r(u), u, j_r(u))\right]$$

and insert u^* between $i_{r'}(u^*)$ and $j_{r'}(u^*)$. This is repeated until all customers are routed or until there are no feasible insertion points for the remaining unrouted customers (no feasible solution).

We must first observe that the definition of the insertion cost formula is the same as the one used by Solomon, that is, a weighted sum of detour (in time units) and delay. However, $c_2(u)$ is quite different since the selection of the next customer to be inserted is based on a generalized regret measure over all routes. The original concept of regret, where the absolute difference between the best and the second best alternative is used as a metric for guiding problem-solving, may be found

in (Tillman and Cain, 1972) for assigning customers to multiple depots in a capacity constrained routing context and in (Martello and Toth, 1981) for solving a generalized assignment problem. The idea to generalize by summing the differences between the best alternative and all the other alternatives is ours and has proven to be useful to foresee difficulties to come with solution feasibility (see below).

Basically, the regret is a kind of 'look ahead' that indicates what can be lost later, if a given customer is not immediately inserted within its best route. In our context, a large regret measure means that there is a large gap between the best insertion place for a customer and its best insertion places in the other routes. Hence, unrouted customers with large regrets must be considered first, since the number of interesting alternative routes for inserting them is small. On the other hand, customers with a small regret measure can be easily inserted into alternative routes without loosing much, and are thus considered later for insertion.

Since $c_1^*(i_r(u), u, j_r(u))$ is set to an arbitrary large value when there is no feasible insertion point within route r for customer u, the generalized regret measure can also foresee difficulties to come with the feasibility of the current solution. In particular, a customer's measure increases by some arbitrary large value whenever a route becomes non feasible for that customer (unless all routes become non feasible for that customer, in which case the metric equals 0). Hence, the generalized regret measure partitions the set of unrouted customers with respect to the number of routes that remain feasible for each one of them. For example, if we assume that the arbitrary large value is L, the regret measure of all unrouted customers with n alternative feasible routes, $1 \leqslant n \leqslant n_r$, is in the neighborhood of $L(n_r - n)$. Since the algorithm looks for the customer with the largest regret measure, it will automatically select a customer with the smallest number of alternative feasible routes, which is the desired behavior. As mentioned before, when the number of feasible routes is the same for two customers, the generalized regret measure then discriminates them by looking at the best insertion places within their feasible routes.

Broadly speaking, this new generalized measure improves upon the classical regret measure by extending the look-ahead to all available alternatives (whereas the classical heuristic is rather 'short sighted' and merely looks at the best and second best alternative). In our context, the generalization of the regret measure has the effect of driving problem-solving more strongly towards feasible solutions.

The parallel approach, as described above, is applied to each problem with three parameter settings

$$(\alpha_1, \alpha_2) = (0.5, 0.5), (0.75, 0.25), (1.0, 0.0),$$

and the best overall solution is chosen. It must however be noted that the algorithm is applied more than three times to each problem because we also look for potential feasible solutions with a smaller number of routes than the one given by Solomon's initial estimate (that is, this estimate is used only as a starting point).

This is done by stopping the procedure as soon as a feasible solution is found for a given number of routes (initially, Solomon's estimate) and a given parameter setting. We then remember the parameter settings that have not yet been tried for the current number of routes (if any), we decrease the number of routes by one and we repeat the whole procedure with $n_r = n_r - 1$ routes. In the process, one of the initial seed customers must be taken away, and this is done by removing the seed customer s^* which is the closest to the other ones, that is

$$\text{remove } s^* = \arg\left[\min_{s \in S_{\text{seed}}} \left\{ \min_{s' \in S_{\text{seed}}, s' \neq s} d_{s,s'} \right\}\right]$$

where S_{seed} is the current set of seed customers.

The number of routes is reduced in this way until we get some value $n_{r_{\text{inf}}}$ for which no feasible solution is found. At that point, we simply backtrack to $n_r = n_{r_{\text{inf}}} + 1$ routes and go on with the remaining parameter settings (if any) for that number of routes. The best solution over the three parameter settings for $n_r = n_{r_{\text{inf}}} + 1$ is then selected as the final solution.

In the very few cases where a feasible solution is not found with the initial n_r estimate, as provided by Solomon's algorithm, additional routes are incrementally added until such a feasible solution is found. When we add a new route, a new seed customer must be chosen and this is done by selecting the customer which is the far-

thest from the current set of seed customers, that is

$$\text{add } u^* = \arg\left[\max_u \left\{\min_{s \in S_{\text{seed}}} d_{s,u}\right\}\right]$$

where S_{seed} is the current set of seed customers.

Note that a pseudo-code description for that algorithm may be found in the Appendix at the end of the paper.

In order to evaluate the potential of the parallel approach and compare it to the sequential approach, we performed a number of computational tests based on Solomon's standard set of problems. Those results are presented in the next section.

4. Computational results

For the computational tests, we used the standard set of problems of Solomon which are all 100-customer euclidean problems. The geographical data are either randomly generated using a uniform distribution (problem sets R1 and R2), clustered (problem sets C1 and C2) or semi-clustered with a mix of randomly distributed and clustered customers (problem sets RC1 and RC2). Problem sets R1, C1 and RC1 have a short scheduling horizon, and allow only a few customers per route. Conversely, problem sets R2, C2 and RC2 have a long scheduling horizon and allow a larger number of customers per route. More details about these problems, in particular details relating to the characteristics of the time windows, may be found in Solomon's paper (Solomon, 1987).

In Section 4.1, we compare results that were obtained with the parallel algorithm and Solomon's sequential algorithm, without any post-optimization procedures (as in Solomon's original paper). Section 4.2 then gives some insight about solution quality with respect to optimum solutions, with and without post-optimization.

4.1. Sequential vs. parallel approach

For these experiments, we ran Solomon's original code as well as our FORTRAN implementation of the parallel algorithm on a IBM-PC compatible microcomputer. Table 1 compares Solomon's sequential approach (Sequential) with the parallel approach (Parallel). For each type of problems, we show the average number of routes, total route time, and computation time (in seconds). The total route time is the sum of total travel time, waiting time and service time.

For Solomon's algorithm, we also show within parentheses the results as reported in his original paper. The slight difference between his results and ours stem from the distinct hardware implementations. In particular, our microcomputer's floating point precision is not the same as the DEC-10 of Solomon, and we observed on a small number of problems that this fact was responsible for slightly different solutions.

For comparison purposes, solution quality is based first on the number of routes, then on total route time.

From the results shown in Table 1, we can draw the following conclusions:

(a) For randomly generated problems, the parallel approach slightly outperformed the se-

Table 1a
Random problems

		Number of routes	Route time	Computation time (sec.)
R1, 12 problems	Sequential	13.5	2688.5	543.5
		(13.6)	(2695.5)	
	Parallel	13.3	2696.0	882.2
R2, 11 problems	Sequential	3.3	2555.0	3332.6
		(3.3)	(2578.1)	
	Parallel	3.1	2513.3	1884.5

Table 1b
Clustered problems

		Number of routes	Route time	Computation time (sec.)
C1, 9 problems	Sequential	10.0	10094.0	577.7
		(10.0)	(10104.2)	
	Parallel	10.7	10610.3	856.9
C2, 8 problems	Sequential	3.1	9918.3	1858.4
		(3.1)	(9921.4)	
	Parallel	3.4	10477.6	1042.7

quential approach. For both problem sets R1 and R2, the total number of routes has been reduced. On problem set R2, the total route time has also been reduced.

(b) For clustered problems C1 and C2, the sequential approach outperformed the parallel approach. This is not too surprising, since a sequential approach is well adapted to such purely clustered problems. We can describe its general behavior in the following way: first, it selects a seed customer in one cluster and 'empties' it during the insertion phase. Then, it initializes a second route by selecting the seed customer in another cluster and empties it again. This process is repeated until all clusters are done.

(c) For mixed problems RC1 and RC2, the parallel approach outperformed the sequential approach. On problem set RC2, in particular we were able to reduce the total number of routes by 10% and the total route time by 6%. If we consider that a fixed number of 10 time units per customer is included for service time (for a total of 1000 time units), the improvement with respect to travel time and waiting time is really in the order of 9%, which is quite substantial. For problem set RC1, the parallel approach still outperformed the sequential approach, by allowing a slight reduction in the total number of routes.

To our knowledge, the results for problem sets R1, R2, RC1 and RC2, as obtained with the parallel algorithm, are the best reported so far. The parallel approach thus looks very interesting when some randomness is introduced into the distribution of the customers. The results are particularly interesting for problem sets RC1 and RC2, since real-life problems seldom are purely random or clustered, and usually show more or less well-defined clustered patterns. The parallel algorithm thus appears as an interesting alternative to the classical algorithm of Solomon for such VRSPTW problems.

4.2. Comparisons with optimum solutions

The second set of experiments is aimed at evaluating the accuracy of the above solutions with respect to optimality. To this end, we compared the parallel and sequential algorithms with solutions generated by the column generation technique of Desrochers et al. (1990). The current state of the technology is such that only a few problems from Solomon's set have been

Table 1c
Mixed problems

		Number of routes	Route time	Computation time (sec.)
RC1, 8 problems	Sequential	13.5	2772.2	483.4
		(13.5)	(2775.0)	
	Parallel	13.4	2877.9	891.6
RC2, 8 problems	Sequential	4.0	2982.2	2574.5
		(3.9)	(2955.4)	
	Parallel	3.6	2807.4	1428.7

Table 2
Characteristics of problems

Problem	% of customers with time windows	Average width of time windows
R101	100	10.0
R102	75	10.0
C101	100	60.76
C102	75	61.27
C106	100	156.15
C107	100	180.00
C108	100	243.28

solved optimally (cf. Section 1). Infortunately, no optimum solutions are known for the sets of problems RC1 and RC2, for which the parallel approach performed particularly well. The only known optimum solutions are for problems R101 and R102 in set R1, and C101, C102, C106, C107, C108 in problem set C1. Time window's characteristics of those problems are shown in Table 2.

The computational results are shown in Table 3a and 3b. The first two columns of Table 3a and 3b are the results respectively obtained with the parallel and sequential algorithms, before and after an Or–Opt post-optimization procedure (Or, 1976). Optimum solutions are shown in the last column of both tables. The first number in each cell is the total number of routes and the second number is the total travel time.

Although the results in Table 3 are not conclusive, we can see that Solomon's algorithm performed very well on clustered problems. It found the minimum number of routes in each case and

Table 3a
Parallel solutions vs. Optimum solutions

Problem	Parallel	Parallel and Or–Opt	Optimum
R101	19	19	18
	1873.6	1782.1	1607.0
R102	18	18	17
	1682.5	1549.6	1434.0
C101	10	10	10
	827.3	827.3	827.3
C102	10	10	10
	1092.5	953.3	827.3
C106	11	11	10
	1063.6	1058.9	827.3
C107	11	11	10
	1438.6	1101.1	827.3
C108	11	11	10
	1534.8	1105.6	827.3

Table 3b
Sequential solutions vs. Optimum solutions

Problem	Sequential	Sequential and Or-Opt	Optimum
R101	21	21	18
	1826.0	1693.3	1607.0
R102	19	19	17
	1711.0	1520.2	1434.0
C101	10	10	10
	851.4	827.3	827.3
C102	10	10	10
	964.6	833.6	827.3
C106	10	10	10
	881.0	827.3	827.3
C107	10	10	10
	902.4	827.3	827.3
C108	10	10	10
	853.1	827.3	827.3

reached optimum, after post-optimization, for problems C101, C106, C107 and C108. The parallel algorithm did find the optimum solution for problem C101 (without any post-optimization!) and the minimum number of routes for problem C102, but it generated one additional route for problems C106, C107 and C108. This supports the previous claim about the power of Solomon's sequential approach for purely clustered problems.

The results are not so impressive, however, for purely random problems. For both problems R101 and R102, the parallel and sequential algorithms did not find the minimum number of routes. After post-optimization, however, we can see that total travel time is not far from optimum. Those results indicate that when some randomness is introduced into the distribution of the customers, the parallel and sequential algorithms do not perform so well with respect to optimality. Hence, there is still room for improvement for such problems and this improvement could perhaps be achieved through more clever initialization procedures for both the sequential and parallel approaches.

5. Concluding remarks

This paper has described a parallel route building procedure for vehicle routing problems with time windows. Computational experiments have shown that the parallel approach is better

than the sequential approach for pure random problems and for problems with a mix of randomly distributed and clustered customers. It is however outperformed by the sequential approach on pure clustered problems. The parallel approach thus looks interesting, in particular for real-life problems that are not purely clustered. Comparisons with optimum solutions have also shown that Solomon's algorithm is quite powerful for purely clustered problems. However, this is not true anymore when some randomness is introduced into the distribution of the customers, and interesting research avenues are opened for improving upon the parallel or sequential algorithms in such contexts.

Appendix

Appendix: The parallel tour building procedure

Notation

n_r = number of routes;
$n_{r\,\min}$ = smallest number of routes;
d_{i_r,j_r} = distance (in time units) between customers i_r and j_r in route r;
b_j = service time at customer j;
$b_{u,j}$ = service time at customer j, given that customer u is inserted just before j;
$\text{detour}(i_r, u, j_r)$ = $d_{i_r,u} + d_{u,j_r} - d_{i_r,j_r}$;
$\text{delay}(i_r, u, j_r)$ = $b_{u,j_r} - b_{j_r}$;
$c_{1r}(i_r, u, j_r)$ = $\alpha_1 \cdot \text{detour}(i_r, u, j_r) + \alpha_2 \cdot \text{delay}(i_r, u, j_r)$, $r = 1, \ldots, n_r$;
$c^*_{1r}(i_r(u), u, j_r(u))$ = $\min_{i_r,j_r} c_{1r}(i_r, u, j_r)$, where i_r and j_r are adjacent customers in route r, $r = 1, \ldots, n_r$;
$c_2(u)$ = $\sum_{r \neq r'}[c^*_{1r}(i_r(u), u, j_r(u)) - c^*_{1r'}(i_{r'}(u), u, j_{r'}(u))]$ where $c^*_{1r'}(i_{r'}(u), u, j_{r'}(u)) = \min_r c^*_{1r}(i_r(u), u, j_r(u))$.

Algorithm. *We first identify a feasible solution with the smallest number of routes. We assume here that a feasible solution can be found with the initial n_r estimate as provided by Solomon's algorithm.*

Step 0. Estimate the initial number of routes n_r and the initial set of seed customers S_{seed,n_r} by applying Solomon's sequential algorithm with the parameter setting $(\beta, \alpha_1, \alpha_2) = (1, 1, 0)$. Seed customers are the ones which are the farthest from the depot in each route.

Step 1. Initialize the set of parameters $S_{(\alpha 1,\alpha 2),n_r}$ with $\{(0.5, 0.5), (0.75, 0.25), (1.0, 0.0)\}$.

Step 2. Select a parameter setting (α_1, α_2) in $S_{(\alpha 1,\alpha 2),n_r}$ and remove it from that set.

Step 3. Call Build-routes $(n_r, \alpha_1, \alpha_2, S_{\text{seed},n_r})$.

Step 4. If Build-routes returns a solution:
remember that solution as the best known solution,
set n_r to $n_r - 1$,
set S_{seed,n_r} to $S_{\text{seed},n_r+1} - s^*$ where:
$s^* = \arg[\min_{s \in S_{\text{seed}},n_r+1}\{\min_{s' \in S_{\text{need}},n_r+1,\ s' \neq s} d_{s,s'}\}]$,
go to Step 1,
otherwise:
if no parameter setting remains in $s_{(\alpha 1,\alpha 2),n_r}$:
set $n_{r\,\min}$ to $n_r + 1$,
go to Step 5,
otherwise:
go to Step 2.

When we reach that point, we know what $n_{r\,\min}$ is, and we now want to identify the best feasible solution with $n_{r\,\min}$ routes.

Step 5. If no parameter setting remains in $S_{(\alpha 1,\alpha 2),n_{r\min}}$:
terminate with the best known solution.

Step 6. Select a parameter setting (α_1, α_2) in $S_{(\alpha 1,\alpha 2),n_{r\min}}$ and remove it from that set.

Step 7. Call Build-routes $(n_{r\min}, \alpha_1, \alpha_2, S_{\text{seed},n_{r\min}})$.
If Build-routes returns a solution:
compare that solution with the best known solution. If the current solution is better, remember it as the best known solution.

Step 8. Go to Step 5.

Procedure Build-routes $(n_r, \alpha_1, \alpha_2, S_{\text{seed},n_r})$

Step 0. Initialize n_r routes, with the n_r customers in the set S_{seed,n_r}.

Step 1. For each unrouted customer u, evaluate feasible insertion costs $c_{1r}(i_r, u, j_r)$ for all adjacent customers i_r and j_r in route r and find $c_{1r}^*(i_r(u), u, j_r(u))$, $r = 1, \ldots, n_r$. Note that $c_{1r}(i_r(u), u, j_r(u))$ is set to an arbitrary large value if there is no feasible insertion point.

Step 2. Select the next customer u^* to be inserted as the one for which $c_2(u)$ is maximum.

Step 3. Consider the insertion of customer u^* between $i_{r'}(u^*)$ and $j_{r'}(u^*)$ in route r', where

$$c_{1r'}^*(i_{r'}(u^*), u^*, j_{r'}(u^*)) = \min_{r=1,\ldots,n_r} c_{1r}^*(i_r(u^*), u^*, j_r(u^*)).$$

If this insertion is feasible:
insert u^*;
if unrouted customers remain:
go to Step 1,
otherwise:
return(solution)
otherwise:
return(NIL).

Acknowledgements

We would like to thank Lucie Bibeau and Patrice Bergeron for the hours spent at running and analyzing the computational experiments. Thanks also to Jacques Desrosiers and Martin Desrochers who gave us the code and the set of problems of Marius M. Solomon, as well as known optimum solutions. Finally, this research would not have been possible without the financial support of the Natural Sciences and Engineering Council of Canada (NSERC).

References

Bodin, L., Golden, b.L., Assad, A., and Ball, M. (1983), "Routing and scheduling of vehicles and crews: The state of the art", *Computers and Operations Research* 10, 62–212.

Christofides, N., Mingozzi, R., and Toth, P. (1979), "The vehicle routing problem", in: N. Christofides, R. Mingozzi, P. Toth and C. Sandi (eds.), *Combinatorial Optimization*, John Wiley, New-York.

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

Desrochers, M., Lenstra, J.K., Savelsbergh, J.K., and Soumis, F. (1988), "Vehicle routing with time windows: Optimization and approximation", in: B.L. Golden and A.A. Assad (eds.), *Vehicle Routing: Methods and Studies*, North-Holland, Amsterdam, 65–84.

Desrochers, M., Desrosiers, J., and Solomon, M. (1990), "Using column generation to solve the vehicle routing problem with time windows", Technical Report G-90-08, GERAD, École des Hautes Études Commerciales, Université de Montréal.

Martello, S., and Toth, P. (1981), "An algorithm for the generalized assignment problem", in: J.P. Brans (ed.), *Operational Research '81: Proceedings of the Ninth IFORS Int. Conf. on Operational Research*, North-Holland, Amsterdam, 589–603.

Mole, R., and Jameson, S. (1976), "A sequential route-building algorithm employing a generalized savings criterion", *Operational Research Quarterly* 27, 503–511.

Or, I. (1976), "Traveling salesman-type combinatorial problems and their relation to the logistics of blood banking", Ph.D. Thesis, Dept. of Industrial Engineering and Management Sciences, Northwestern University.

Potvin, J.Y. and Rousseau, J.M. (1990), "Computational experiments with exchange procedures for the multiple traveling salesmen problem with time windows", Technical Report #729, Centre de Recherche sur les Transports, Université de Montréal.

Savelsbergh, M.W.P. (1990), "An efficient implementation of local search algorithms for constrained routing problems", *European Journal of Operational Research* 47, 75–85.

Solomon, M.M. (1987), "Algorithms for the vehicle routing and scheduling problems with time window constraints", *Operations Research* 15/2, 254–265.

Solomon, M.M., and Desrosiers, J. (1988), "Time window constrained routing and scheduling problems", *Transportation Science* 22/1, 1–13.

Solomon, M.M., Baker, E.K., and Schaffer, J.R. (1988), "Vehicle routing and scheduling problems with time window constraints: Efficient implementations of solution improvement procedures", in: B.L. Golden and A.A. Assad (eds.), *Vehicle Routing: Methods and Studies*, North-Holland, Amsterdam, 85–105.

Sorensen, B. (1986), "Interactive distribution planning", Ph.D. Dissertation, IMSOR, Lyngby, Denmark.

Tillman, F., and Cain, T. (1972), "An upper bounding algorithm for the single and multiple terminal delivery problem", *Management Science* 18, 664–682.