A sequential insertion heuristic for the initial solution to a constrained vehicle routing problem

JW Joubert[*]   SJ Claasen[†]

Abstract

The Vehicle Routing Problem (VRP) is a well-researched problem in the Operations Research literature. It is the view of the authors of this paper that the various VRP variants have been researched in isolation. This paper embodies an attempt to integrate three specific variants of the VRP, namely the VRP with multiple time windows, the VRP with a heterogeneous fleet, and the VRP with double scheduling, into an initial solution algorithm. The proposed initial solution algorithm proves feasible for the integration, while the newly introduced concept of time window compatibility decreases the computational burden when using benchmark data sets from literature as a basis for efficiency testing. The algorithm also improves the quality of the initial solution for a number of problem classes.

1   Introduction

The Vehicle Routing Problem (VRP) is a well-researched problem in the Operations Research literature. The main objective in this type of problem is to minimize an objective function value, which is typically distribution cost for individual carriers. The area of application is wide, and specific variants of the VRP transform the basic problem to conform to application specific requirements. It is the view of the authors that the various VRP variants have been researched in isolation, with little effort to integrate various problem variants into an instance that is more appropriate to the South African particularity with regards to logistics and vehicle routing.

The VRP may be described as the problem of assigning optimal delivery or collection routes from a depot to a number of geographically distributed customers, subject to side constraints. The most basic version of the VRP may be defined in terms of a bi-directed complete graph $G = (V, E)$, where $V = \{v_0, v_1, \ldots, v_n\}$ is a set of vertices with v_0 representing the depot where m identical vehicles, each with capacity Q, are located. The

---

[*]Corresponding author: Department of Industrial and Systems Engineering, University of Pretoria, Pretoria, 0002, South Africa, email: `johan.joubert@up.ac.za`

[†]Department of Industrial and Systems Engineering, University of Pretoria, Pretoria, 0002, South Africa

remaining vertices, denoted by $V \backslash \{v_0\}$, represent customers each having a non-negative demand q_i and a non-negative service time s_i [17]. The edge set connecting the vertices is given by $E = \{(v_i, v_j) \mid v_i, v_j \in V, \quad i \neq j\}$. A distance matrix $C = \{c_{ij}\}$ is defined on E. In some contexts, c_{ij} may be interpreted as travel cost or travel distance from vertex v_i to vertex v_j. Hence, the terms distance, travel cost, and travel time are used interchangeably. The VRP consists of designing a set of m vehicle routes having a minimum total length such that:

- each route starts and ends at the depot,

- each remaining vertex ($V \backslash \{v_0\}$) is visited exactly once by exactly one vehicle,

- the total demand of a route does not exceed Q, and

- the total duration (including service and travel time) of a route does not exceed a preset limit L.

The VRP is an NP-hard combinatorial optimization problem for which several exact and approximate solution methods have been proposed (see Laporte [8] for a review). Early researchers, such as Clarke and Wright [1], realized that exact algorithms can only solve relatively small problems, but a number of heuristic algorithms have proved very satisfactory, in many cases yielding near-optimal solutions to relatively large problems.

The basic VRP is based on a number of assumptions, such as utilizing a homogeneous fleet, a single depot, and allocating one route per vehicle. These assumptions may be eliminated or relaxed by introducing additional constraints to the problem. This implies increasing the complexity of the problem, and, by restriction, classifies the extended problem as an NP-hard problem. It should be noted that most of these additional constraints are often implemented in isolation, without integration, due to the increased complexity of solving such problems. Thus the problem statement:

> Is it possible to solve a vehicle routing problem with multiple integrated constraints?

Finding a feasible, and integrated initial solution to a hard problem is the first step in addressing the scheduling issue. In this paper an algorithm is proposed that integrates three specific variants of the VRP. The paper also contributes to reducing the computational burden by proposing a concept referred to as time window compatibility (TWC) to evaluate the insertion of customers on positions within routes intelligently. The authors investigate the feasibility of integrating multiple soft time windows, a heterogeneous fleet and double scheduling constraints into a single problem instance, referred to simply in this paper as the *Vehicle Routing Problem with Multiple Constraints* (VRPMC).

A time window is the period of time during which deliveries can be made to a specific customer, indexed by i, and has three main characteristics: the earliest allowed arrival time denoted by e_i, the latest allowed arrival time denoted by l_i, and whether the time window is considered *soft* (allowing a penalized late service) or *hard* (no late service allowed). It is an extension of the VRP that has been researched extensively [5, 14, 15, 16].

Gendreau *et al.* [3] propose a solution methodology for cases where the fleet is heterogeneous, that is, where the fleet is composed of vehicles with different capacities and costs. Their objective is to determine what the optimal fleet composition should be, and is referred to as either the *Heterogeneous Fleet Vehicle Routing Problem* (HVRP), or the *Fleet Size and Mix Vehicle Routing Problem* (FSMVRP). Taillard [14] formulates the *Vehicle Routing Problem with a Heterogeneous fleet of vehicles* (VRPHE) where the number of vehicles of type t in the fleet is limited; the objective being to optimize the utilization of the given fleet. Salhi and Rand [12] incorporate vehicle routing into the vehicle composition problem, and refer to it as the *Vehicle Fleet Mix* problem (VFM).

Double scheduling occurs where vehicles are routed in a manner that allows a vehicle to complete one route, return to the depot to replenish its capacity, *i.e.* load for deliveries or unload collected cargo, before embarking on a subsequent route. The aggregated routes for a vehicle is referred to as a tour, and a vehicle is required to complete its tour within the depot's provided time window.

The concept of TWC is introduced in §2 along with the initial solution algorithm. The results from simulated data sets are presented in §3, before conclusions are drawn, and a research agenda is established in §4.

2 An initial solution approach

Heuristics typically use a greedy approach to obtain a good initial solution in an efficient manner and then incrementally improve the solution by neighborhood exchanges or local searches. Solomon [13] divides VRP tour-building algorithms into either sequential or parallel methods. Sequential procedures construct one route at a time until all customers are scheduled. Parallel procedures are characterized by the simultaneous construction of routes, while the number of parallel routes may either be limited to a predetermined number, or formed freely. Solomon concludes that, from the five initial solution heuristics evaluated, the *Sequential Insertion Heuristic* (SIH) proved to be very successful, both in terms of the quality of the solution, as well as the computational time required to find the solution [9].

When finding an initial solution to a routing problem, the *initialization criteria* refers to the process of finding the first customer to insert into a route. The most commonly used initialization criteria is the farthest unrouted customer, and the customer with the earliest deadline, or the earliest latest allowed arrival. The first customer inserted on a route is referred to as the *seed customer*. Once the seed customer has been identified and inserted, the SIH algorithm considers, for the unrouted nodes, the insertion place that minimizes a weighted average of the additional distance and time needed to include a customer in the current partially constructed route – referred to as determining the insertion criteria. The third step, the *selection criteria*, tries to maximize the benefit derived from inserting a customer in the current partial route rather than on a new direct route. Note that the terms nodes and customers are used interchangeably. It can easily be shown that the number of criteria calculations for the SIH algorithm is a third order polynomial function of the number of nodes in the network.

A shortcoming of Solomon's SIH [13] is that it considers all unrouted nodes when calculating the insertion and selection criteria for each iteration. The fact that all unrouted nodes are considered makes it computationally expensive. The VRP variant considered in this paper has multiple additional constraints. The occurrence of infeasible nodes, due to their incompatible time windows, in a partially constructed route therefore becomes significant. The introduction of the TWC concept assists in identifying and eliminating the obvious infeasible nodes. This results in a more effective and robust route construction heuristic.

The purpose of TWC is to determine the time overlap of all edges, or node combinations, (v_i, v_j), where $i, j \in \{0, 1, 2, \ldots, n\}$. During the route construction phase, TWC may be tested, and nodes that are obviously infeasible may be eliminated from the set of considered nodes. The *Time Window Compatibility Matrix* (TWCM) is a nonsymmetrical matrix as the sequence of two consecutive nodes, v_i and v_j, is critical. The following notation is used in the problem formulation:

e_i: the earliest allowed arrival time at customer i,

l_i: the latest allowed arrival time at customer i,

s_i: the service time at node i,

t_{ij}: the travel time from node i to node j,

$a_j^{e_i}$: the actual arrival time at node j, given that node j is visited directly after node i, and that the actual arrival time at node i was e_i,

$a_j^{l_i}$: the actual arrival time at node j, given that node j is visited directly after node i, and that the actual arrival time at node i was l_i, and

TWC_{ij}: the TWC when node i is directly followed by node j.

Here TWC_{ij} indicates the entry in row i, column j of the TWCM. Five scenarios exist and are covered in more detail by Joubert [6]. The scenarios depend on the level and direction of overlap between the time windows of two consecutive customers, and are represented in Figure 1.

Each scenario represents a relationship between e_i, l_i, $a_j^{e_i}$ and $a_j^{l_i}$, and assumes customer j to be serviced directly after customer i. In its generalized form, the expression for TWC_{ij} is given by

$$TWC_{ij} = \begin{cases} \min\left\{a_j^{l_i}, l_j\right\} - \max\left\{a_j^{e_i}, e_j\right\} & \text{if } l_j - a_j^{e_i} > 0, \\ -\infty & \text{otherwise.} \end{cases} \tag{1}$$

The higher the value of TWC_{ij} in (1), the better the compatibility of the two time windows considered. Therefore an incompatible time window is defined to have a compatibility of negative infinity.

Consider an improved case where node v_u is considered for insertion between nodes v_i and v_j. As the TWCM is already calculated, it is possible to test the compatibility of node v_u with the routed nodes v_i and v_j. If either TWC_{iu} or TWC_{uj} is negative infinity $(-\infty)$, indicating an incompatible time window, the insertion heuristic moves on and considers the next edge, without wasting computational effort on calculating the insertion and selection criteria. Only if the time windows are considered compatible will the insertion and selection criteria be evaluated. The improvement of the computational burden is a direct function
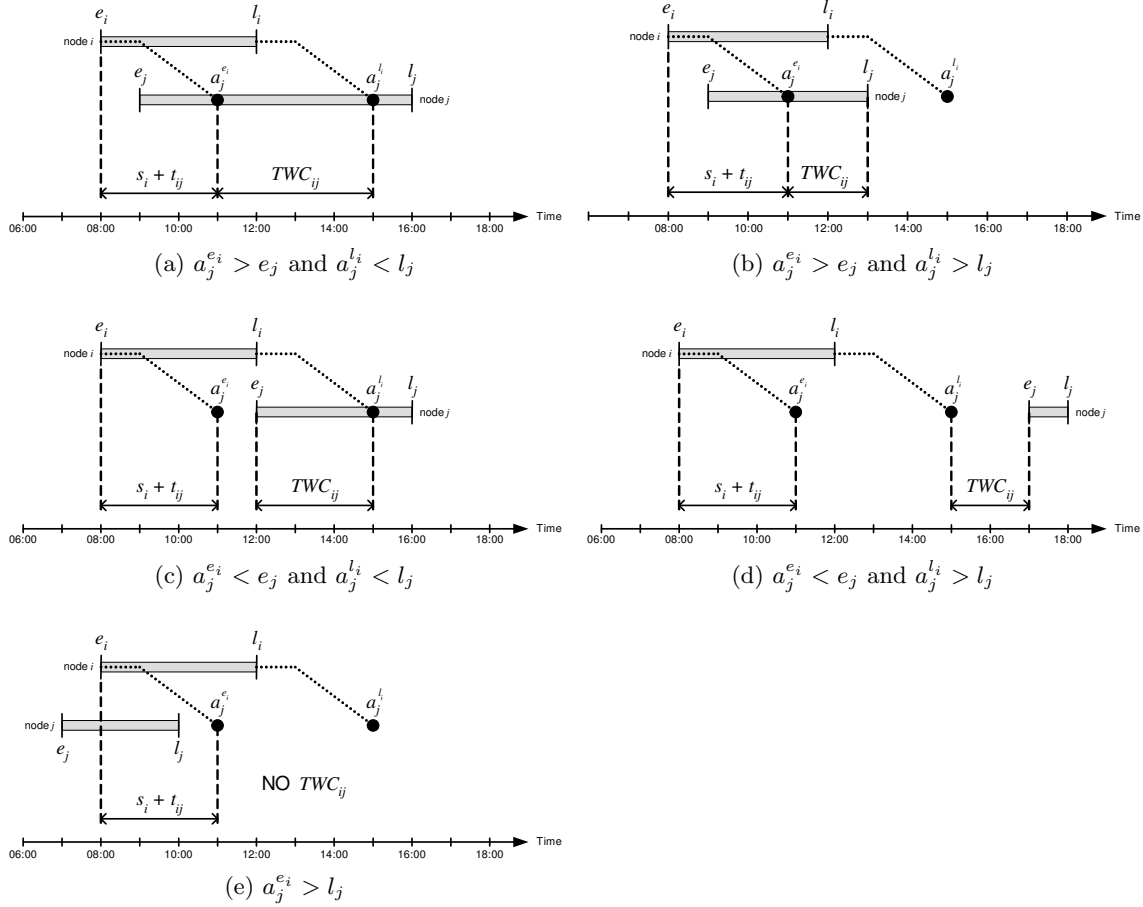
Figure 1: Time window compatibility scenarios.

of the characteristics of the customer time windows. The computational complexity is of the same order as that of the SIH, with expected constant factor improvements possible due to the TWCM.

As opposed to the two most common initialization criteria, namely *customer with earliest deadline*, and *furthest customer*, as suggested by Dullaert *et al.* [2], the authors of this paper also use the TWCM to identify seed nodes based on their TWC (the number of infeasible TWCs is calculated for each customer). The customer with the highest number of TWCs is identified as the seed customer. Ties are broken arbitrarily. It may be possible to not have any infeasible time window instances. In such a case a total compatibility value may be determined for each node v_a by means of the expression

$$\sum_{\substack{i=1 \\ i \neq a}}^{M} TWC_{ia} + \sum_{\substack{j=1 \\ j \neq a}}^{M} TWC_{aj} + TWC_{aa}, \tag{2}$$

where M denotes the number of unrouted nodes. The customer with the lowest total compatibility is selected as seed customer. A graphical presentation of the initial solution algorithm is given in Figure 2.
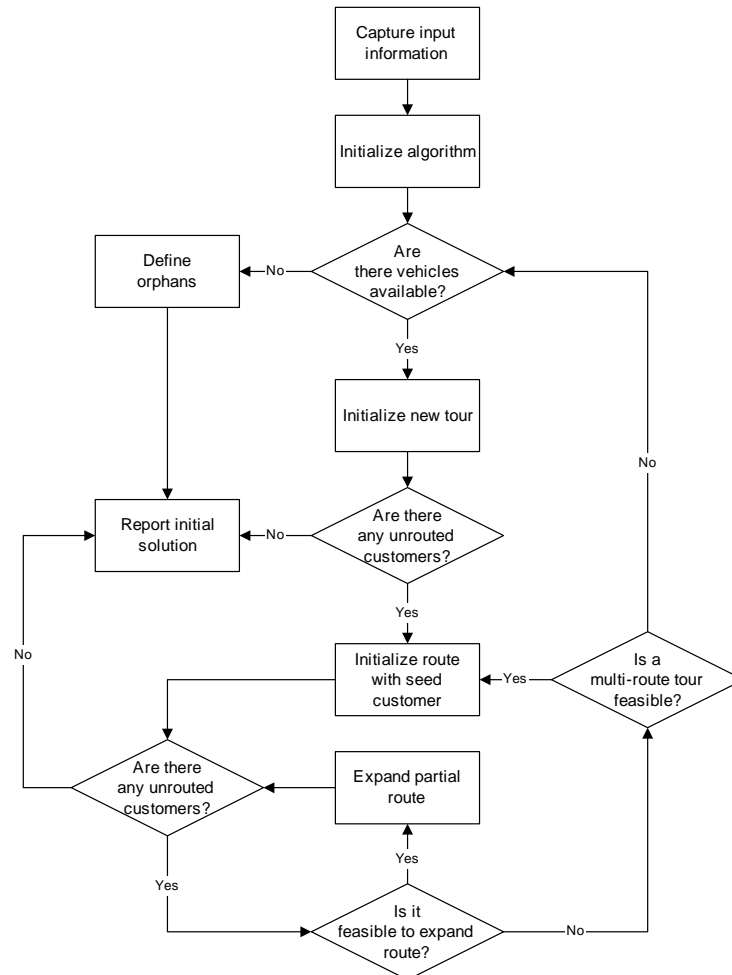
Figure 2: Overview of the initial solution algorithm.

3   Results

The algorithm's objective function is defined as the total scheduling distance. For algorithmic evaluation purposes the basic concept of Solomon [13] is used in terms of classifying benchmark data sets as being either clustered (C), randomly distributed (R), or a combination of the two (RC), and having either short, or long scheduling horizons.

To incorporate multiple time windows, an extended data set from Homberger [4] is used. The data set was generated in a similar fashion to that of Solomon [13], but contains 200 customers. The time windows from customers 101 through 200 were used as second time window for customers 1 through 100. Where an overlap of time windows occurred, a single (wider) time window was created by combining the two overlapping windows. The earliest opening time of the two time windows is used along with the latest closing time [7].

The fleet structure proposed by Liu and Shen [11, 10] is used to introduce their insertion based savings heuristic, and to incorporate a heterogeneous fleet. It is important to evaluate the contribution that the proposed TWC has on the results of the algorithm.

For this purpose a comparative control algorithm is created. The control algorithm differs only in two respects from the proposed algorithm:

- It does not evaluate nodes for TWC when calculating the insertion criteria, and therefore considers every node for insertion on every edge of a partially constructed route.

- As no TWC is calculated for any node, the initialization criterium is changed to identify the seed customer as the unrouted customer with the earliest deadline.

The control algorithm was executed for all problem classes, and the initial solution summaries are provided in Table 1.

Although the results from the various problem classes, and even among the instances within a problem class, vary significantly, there is sufficient evidence that the notion of TWC improves the quality of the initial solution, on average from 60 problem instances, by more than 9%. There is, in general, a direct relationship between the distance saving and the computational saving for the proposed algorithm. It may be observed that when there are distance savings, there are also computational savings.

Although the specific problem instance impacts both the quality of the solution found, and the time required to find such a solution, it is clear that when customers with tight time windows are either randomly distributed or distributed in a semi-clustered manner, that the proposed algorithm performs consistently worse than the control algorithm, both in terms of finding a good quality solution, and in the computational time required. Should the $R1$ and $RC1$ problem classes be omitted, *i.e.* the proposed algorithm not be applied to such problem instances, then an average distance saving of 14%, and an average CPU time saving of 19% may be achieved, respectively.

4   Conclusions

The results of the initial solution algorithm proposed in this paper demonstrate that multiple variants of the vehicle routing problem, *i.e.* multiple soft time windows, a heterogeneous fleet and double scheduling, may indeed be integrated into a single initial solution algorithm.

The initial solution algorithm is the first step in obtaining near optimal solutions for vehicle routing and scheduling type problems. In this paper the concept of *time window compatibility* is introduced to ease the computational burden of the algorithm, and to find a better seed customer. The amount by which the proposed algorithm eases the computational burden is a direct function of the time window characteristics of the customers in the network. Numerical results indicate that initial solution algorithms are highly sensitive to the specific problem instance. This paper claims that the proposed algorithm holds both distance and savings opportunities for problem instances where customers are clustered, or where longer time windows exist for customers.

Problem class	TWC included	Number of tours	Number of routes	Scheduling distance	Distance saving	CPU time saving
C1-01	✓	11	22	7 330	18%	43%
	✗	11	39	8 914		
C1-02	✓	12	23	7 014	12%	20%
	✗	11	33	8 016		
C1-03	✓	16	21	5 626	−1%	0%
	✗	14	24	5 566		
C1-04	✓	17	19	6 318	−26%	−24%
	✗	17	21	5 020		
C1-05	✓	14	21	6 110	30%	33%
	✗	11	35	8 668		
C1-06	✓	12	21	7 280	9%	34%
	✗	11	30	8 020		
C1-07	✓	18	19	6 462	18%	36%
	✗	9	27	7 886		
C1-08	✓	17	19	6 134	18%	28%
	✗	11	24	7 518		
C1-09	✓	18	19	5 842	18%	23%
	✗	10	21	7 142		
C1-10	✓	18	19	5 434	21%	7%
	✗	12	19	6 838		

Table 1a: Results for the *C*1-class of problems.

Problem class	TWC included	Number of tours	Number of routes	Scheduling distance	Distance saving	CPU time saving
C2-01	✓	4	6	7 240	0%	32%
	✗	4	19	7 246		
C2-02	✓	4	8	5 820	35%	16%
	✗	4	20	8 968		
C2-03	✓	5	9	5 744	19%	18%
	✗	5	13	7 052		
C2-04	✓	5	6	5 138	5%	−24%
	✗	4	7	5 410		
C2-05	✓	4	6	7 152	15%	19%
	✗	4	20	8 382		
C2-06	✓	4	6	6 478	22%	−3%
	✗	4	17	8 294		
C2-07	✓	3	7	6 608	8%	24%
	✗	4	14	7 148		
C2-08	✓	4	6	5 874	18%	17%
	✗	3	12	7 186		
C2-09	✓	5	7	5 966	7%	22%
	✗	4	13	6 438		
C2-10	✓	4	6	6 088	26%	13%
	✗	4	10	8 220		

Table 1b: Results for the *C*2-class of problems.

Problem class	TWC included	Number of tours	Number of routes	Scheduling distance	Distance saving	CPU time saving
R1-01	✓	16	72	16 178	−4%	−56%
	✗	7	75	15 510		
R1-02	✓	40	69	11 216	−1%	−31%
	✗	30	70	11 068		
R1-03	✓	56	69	10 770	−0%	−15%
	✗	53	69	10 730		
R1-04	✓	51	70	15 498	−2%	−34%
	✗	44	71	15 150		
R1-05	✓	43	68	11 430	2%	−60%
	✗	7	71	11 606		
R1-06	✓	47	68	10 712	3%	−32%
	✗	27	69	11 062		
R1-07	✓	60	69	10 618	0%	−24%
	✗	49	68	10 640		
R1-08	✓	56	70	16 076	−1%	−35%
	✗	44	71	15 898		
R1-09	✓	41	68	10 816	2%	−50%
	✗	8	69	11 042		
R1-10	✓	47	69	10 756	5%	63%
	✗	10	68	11 362		

Table 1c: Results for the $R1$-class of problems.

Problem class	TWC included	Number of tours	Number of routes	Scheduling distance	Distance saving	CPU time saving
R2-01	✓	5	7	8 722	13%	53%
	✗	2	13	9 990		
R2-02	✓	4	6	6 162	29%	33%
	✗	2	13	8 636		
R2-03	✓	2	6	5 412	15%	23%
	✗	2	9	6 336		
R2-04	✓	3	6	3 474	13%	2%
	✗	2	6	3 988		
R2-05	✓	5	6	7 874	27%	51%
	✗	2	9	10 792		
R2-06	✓	3	6	5 520	30%	31%
	✗	2	8	7 848		
R2-07	✓	4	7	4 544	37%	19%
	✗	2	9	7 228		
R2-08	✓	5	6	3 544	25%	−3%
	✗	1	6	4 722		
R2-09	✓	5	6	7 320	11%	50%
	✗	2	8	8 198		
R2-10	✓	5	7	7 792	20%	45%
	✗	2	8	9 768		

Table 1d: Results for the $R2$-class of problems.

Problem class	TWC included	Number of tours	Number of routes	Scheduling distance	Distance saving	CPU time saving
RC1-01	✓	26	48	8 706	4%	−6%
	×	6	50	9 026		
RC1-02	✓	32	45	8 662	−1%	1%
	×	13	46	8 542		
RC1-03	✓	39	46	8 882	−7%	−33%
	×	26	47	8 268		
RC1-04	✓	45	46	10 368	−30%	−103%
	×	45	47	7 960		
RC1-05	✓	34	47	8 754	4%	−10%
	×	10	49	9 142		
RC1-06	✓	40	47	8 374	11%	−16%
	×	6	48	9 418		
RC1-07	✓	41	47	8 440	5%	−13%
	×	10	48	8 846		
RC1-08	✓	41	47	8 702	0%	−17%
	×	12	49	8 742		
RC1-09	✓	40	47	8 730	8%	−26%
	×	11	48	9 444		
RC1-10	✓	38	48	8 630	0%	−18%
	×	14	48	8 630		

Table 1e: Results for the *RC*1-class of problems.

Problem class	TWC included	Number of tours	Number of routes	Scheduling distance	Distance saving	CPU time saving
RC2-01	✓	12	19	7 748	10%	24%
	×	2	21	8 622		
RC2-02	✓	9	18	6 938	7%	22%
	×	2	20	7 480		
RC2-03	✓	10	18	5 914	−14%	7%
	×	3	18	5 208		
RC2-04	✓	16	18	6 512	−19%	−28%
	×	14	18	5 452		
RC2-05	✓	15	19	7 730	2%	27%
	×	3	19	7 892		
RC2-06	✓	16	19	6 886	12%	22%
	×	3	20	7 788		
RC2-07	✓	18	19	6 362	16%	23%
	×	3	18	7 612		
RC2-08	✓	14	18	5 832	18%	13%
	×	3	19	6 936		
RC2-09	✓	15	18	5 748	19%	16%
	×	3	19	7 114		
RC2-10	✓	16	18	4 804	27%	14%
	×	2	18	6 622		

Table 1f: Results for the *RC*2-class of problems.

At worst, the number of evaluative criteria calculations is a third order polynomial function of the number of nodes in the network, similar to the SIH on which it is based. Currently the TWC is determined after the selection criteria, and future research could evaluate the positioning of the TWC portion of the algorithm.

As the proposed algorithm is but the first step in obtaining a final solution, integration with metaheuristics is required before the algorithm can be implemented. Some metaheuristic algorithms, such as Tabu Search, are sensitive to the quality of an initial solution. The concept of TWC should again be introduced and evaluated at metaheuristic level as a potential performance improvement tool.

Acknowledgement

References

[1] CLARKE G & WRIGHT JW, 1964, *Scheduling of vehicles from a central depot to a number of delivery points*, Operations Research, **12**, pp. 568–581.

[2] DULLAERT W, JANSSENS GK, SÖRENSEN K & VERNIMMEN B, 2001, *New heuristics for the fleet size and mix vehicle routing problem with time windows*, in 9th World Conference on Transport Research, COEX Convention Center, Seoul.

[3] GENDREAU M, LAPORTE G, MUSARAGANYI C & TAILLARD ÉD, 1999, *A tabu search heuristic for the heterogeneous fleet vehicle routing problem*, Computers & Operations Research, **26(12)**, pp. 1153–1173.

[4] HOMBERGER J & GEHRING H, 1999, *Two evolutionary metaheuristics for the vehicle routing problem with time windows*, Information Systems and Operations Research, **37(3)**, pp. 297–318.

[5] IBARAKI T, IMAHORI S, KUBO M, MASUDA T, UNO T & YAGIURA M, 2005, *Effective local search algorithms for routing and scheduling problems with general time window constraints*, Transportation Science, **39(2)**, pp. 206–232.

[6] JOUBERT JW, 2003, *An initial solution heuristic for the vehicle routing and scheduling problem*, MEng thesis, University of Pretoria, Pretoria.

[7] JOUBERT JW, 2004, *Improving on the initial solution heuristic for the vehicle routing problem with multiple constraints*, pp. 217–226 in BREBBIA C (ED), *Urban Transport X*, WITPress, Southampton.

[8] LAPORTE G, 1992, *The vehicle routing problem: An overview of exact and approximate algorithms*, European Journal of Operational Research, **59(3)**, pp. 345–358.

[9] LAPORTE G & SEMET F, 2002, *Classical heuristics for the capacitated VRP*, pp. 109–154 in TOTH P & VIGO D (EDS), *The vehicle routing problem*, SIAM, Philadelphia (PA).

[10] Liu F-H & Shen S-Y, 1999, *The fleet size and mix vehicle routing problem with time windows*, Journal of the Operational Research Society, **50(7)**, pp. 721–732.

[11] Liu F-H & Shen S-Y, 1999, *A method for vehicle routing problem with multiple vehicle types and time windows*, Proceedings of the National Science Council, Republic of China, ROC(A), **23(4)**, pp. 526–536.

[12] Salhi S & Rand GK, 1993, *Incorporating vehicle routing into the vehicle fleet composition problem*, European Journal of Operational Research, **66(3)**, pp. 313–330.

[13] Solomon MM, 1987, *Algorithms for the vehicle routing and scheduling problems with time windows*, Operations Research, **35(2)**, pp. 254–265.

[14] Taillard ÉD, 1999, *A heuristics column generation method for the heterogeneous fleet* VRP, Operations Research — Recherche opérationnelle, **33**, pp. 1–14.

[15] Taillard ÉD, Badeau P, Gendreau M, Guertin F & Potvin J-Y, 1997, *A tabu search heuristic for the vehicle routing problem with soft time windows*, Transportation Science, **31(2)**, pp. 170–186.

[16] Tan KC, Lee LH, Zhu QL & Ou K, 2001, *Heuristic methods for vehicle routing problem with time windows*, Artificial Intelligence in Engineering, **15(3)**, pp. 281–295.

[17] Van Breedam A, 2001, *Comparing descent heuristics and metaheuristics for the vehicle routing problem*, Computers & Operations Research, **28(4)**, pp. 289–315.