Transportation Research Part E 48 (2012) 616–636

Contents lists available at SciVerse ScienceDirect

Transportation Research Part E

journal homepage: www.elsevier.com/locate/tre

The time dependent vehicle routing problem with time windows: Benchmark problems, an efficient solution algorithm, and solution characteristics

Miguel Andres Figliozzi

Maseeh College of Engineering and Computer Science, PO Box 0751, Portland State University, Portland, OR 97207-0751, United States

ARTICLE INFO

Article history:
Received 15 August 2011
Received in revised form 23 September 2011
Accepted 6 November 2011

Keywords:
Vehicle routing
Benchmark problems
Time dependent travel time
Variable speed
Congestion

ABSTRACT

An algorithm that can tackle time dependent vehicle routing problems with hard or soft time windows without any alteration in its structure is presented. Analytical and experimental results indicate that average computational time increases proportionally to the number of customers squared. New replicable test problems that capture the typical speed variations of congested urban settings are proposed. Solution quality, time window perturbations, and computational time results are discussed as well as a method to study the impact of perturbations by problem type. The algorithm efficiency and simplicity is well suited for urban areas where fast running times may be required.

1. Introduction

Congestion is a common phenomenon in most urban areas of the world. Congestion creates a substantial variation in travel speeds during peak morning and evening hours. This is problematic for all vehicle routing models that rely on a constant value to represent vehicle speeds. Urban route designs that ignore these significant speed variations result in inefficient and suboptimal solutions. Poorly designed routes that lead freight vehicles into congested arteries and streets not only increase supply chain and logistics costs but also worsen externalities associated with freight traffic in urban areas such as greenhouse gases, noise, and air pollution. Travel time between customers and depot is found to be a crucial factor that amplifies the negative impacts of congestion; congestion also affects carriers' cost structure and the relative weight of wages and overtime expenses (Figliozzi, 2010a).

Routing models with time-varying travel times are gaining greater attention in vehicle routing literature and industry. However, research on the time dependent vehicle routing problem (TDVRP) is still comparatively meager in relation to the body of literature accumulated for the classical vehicle routing problem (VRP) and vehicle routing problem with time windows (VRPTW). In addition, published algorithms and related results can neither be readily benchmarked nor do they cover all practical and relevant objective functions or time window constraint types. Without readily available benchmark problems the solution quality of TDVRP algorithms cannot be properly analyzed. Furthermore, both solution quality and computation times are key attributes of a desirable TDVRP algorithm. Fast algorithms are particularly critical in urban TDVRP because travel time updates and new solutions may be necessary due to non-recurrent events such as accidents.

The goals of this research are to: (a) formulate a time dependent vehicle routing problem with a general cost function and time window constraints, (b) present an intuitive and efficient solution methodology for problems with time dependent speeds, (c) introduce readily replicable time dependent instances and analyze the computational results and robustness

E-mail address: figliozzi@pdx.edu

1366-5545/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.tre.2011.11.006

of the solutions, and (d) analyze the computational complexity of the solution approach. This paper is organized as follows: Section 2 presents a literature review for the TDVRP; Section 3 introduces notation and formulates the problems. Section 4 presents the Iterative Route Construction and Improvement algorithm (IRCI) to solve time dependent routing problems and minimize fleet size; Section 5 presents algorithms to reduce soft time window penalties and route durations; Section 6 presents benchmark problems; Section 7 discusses computational results; Section 8 analyses the worst case and average computational complexity of the algorithms presented in Sections 4 and 5; and Section 9 concludes the paper.

2. Literature review

Unlike widely studied versions of the VRP, e.g. capacitated VRP or time windows VRP with constant travel speed or no travel time variability, time dependent problems have received considerably little attention. The time dependent VRP was first formulated by Malandraki (1989) and Malandraki and Daskin (1992) using a mixed integer linear programming formulation. A greedy nearest-neighbor heuristic based on travel time between customers was proposed, as well as a branch and cut algorithm to solve TDVRP *without* time windows. Hill and Benton (1992) considered a node based time dependent vehicle routing problem (without time windows). Computational results for one vehicle and five customers were reported. Ahn and Shin (1991) discussed modifications to the savings, insertion, and local improvement algorithms to better deal with TDVRP. In randomly generated instances, they reported computation time reductions as a percentage of "unmodified" savings, insertion, and local improvement algorithms. Malandraki and Dial (1996) proposed a dynamic programming algorithm for the time dependent traveling salesman problem, i.e. for a fleet of just one vehicle. A nearest-neighbor type heuristic was used to solve randomly generated problems.

An important property for time dependent problems is the First In–First Out (FIFO) property (Ahn and Shin, 1991; Ichoua et al., 2003). A model with a FIFO property guarantees that if a vehicle leaves customer i to go to customer j at any time t, any identical vehicle with the same destination leaving customer i at a time $t + \varepsilon$, where $\varepsilon > 0$, will always arrive later. This is an intuitive and desirable property though it is not present in all models. Earlier formulations and solutions methods, Malandraki (1989), Malandraki and Daskin (1992), Hill and Benton (1992), and Malandraki and Dial (1996), do not guarantee the FIFO property as reported by Ichoua et al. (2003). Later research efforts have modeled travel time variability using "constant speed" time periods which guarantees the FIFO property, as shown by Ichoua et al. (2003).

Ichoua et al. (2003) proposed a tabu search solution method, based on the work of Taillard et al. (1997), in order to solve time dependent vehicle routing problems with *soft* time windows. Ichoua et al. showed that ignoring time dependency, i.e. using VRP models with constant speed, can lead to poor solutions. Ichoua et al. tested their method using the Solomon problem set, soft time windows, three time periods, and three types of time dependent arcs. The objective was to minimize the sum of total travel time plus penalties associated with time window violations. Summarizing, Ichoua et al. solved a different and simpler problem than the one analyzed in this paper. This research deals with a problem with hard time windows whereas Ichoua et al. deals with a problem with soft time windows *only*. Finally, Ichoua et al. assume that the number of routes is known *a priori* and that violations to time window constraints are acceptable. Furthermore, the objective function in Ichoua et al. is to minimize a weighted sum of total distance travelled and total lateness over all customers whereas this research follows the more traditional hierarchical approach used in the vast majority of VRP problems: first minimize number of routes and second minimize time and/or distance.

Other approaches include the work of Fleischmann et al. (2004) who utilized a route construction methods already proposed in the literature, savings and insertion, to solve uncapacitated time dependent VRP with and without time windows. Fleischmann et al. tested their algorithms in instances created from Berlin travel time data. Jung and Haghani proposed a genetic algorithm to solve time dependent problems (Jung and Haghani, 2001; Haghani and Jung, 2005). Using randomly generated test problems, the performance of the genetic algorithm was evaluated by comparing its results with exact solutions (up to 9 customers and 15 time periods) and a lower bound (up to 25 customers and 10 time periods).

More recently Van Woensel et al. (2008) used a tabu search to solve the capacitated vehicle routing problem with time dependent travel times. To determine travel speed, approximations based on queuing theory and the volumes of vehicles in a link were used. Van Woensel et al. solved capacitated VRP (with no time windows) for problem sizes between 32 and 80 customers. Donati et al. (2008) proposed a solution adapting the ant colony heuristic approach and a local search improvement approach that stores and updates the slack times or feasible delays. The heuristic was tested using a real life network in Padua, Italy, and some variations of the Solomon problem set. The latest work includes a paper by Soler et al. (2009) who proposed a method to solve, optimally, TDVRP instances that are too small for practical purposes and likely exponential growth of computational time as a function of problem size. Dabia et al. (2010) deals with a one-vehicle vehicle routing problem using a dynamic programming approach. Kok (2010) deals with the TDVRP with a focus on departure time optimization and driver break scheduling. This work actually uses a modification of the set of benchmark instances for the VRP with time-dependent travel speeds proposed by an early working paper by Figliozzi (2009). In addition to the work of Kok, only two papers use well known benchmark problems with time windows. Ichoua et al. (2003) used the widely known Solomon problems for the VRP with time windows. However, capacity constraints were not considered, optimal fleet size was given, and no details were provided regarding how links were associated with "categories" that represent differences in the urban network (i.e. main arteries, local streets, etc.). Donati et al. (2008) also used Solomon instances, however, the results cannot be compared with previous results by Ichoua et al. (2003) because a different time speed function was used and capacity constraints were considered. In addition, the exact instances used by Donati et al. (2008) cannot be reconstructed because

the different travel speeds were *randomly* assigned to arcs. Therefore, no study published to date can be swiftly replicated and solution qualities and computation times cannot be compared.

Comparisons are also problematical because objective functions and routing constraints for time dependent problems are often dissimilar, unlike VRPTW research where the objective function is hierarchical and usually considers fleet size (primary objective), distance (secondary objective), and total route duration. Ichoua et al. (2003) study the TDVRP with soft time windows and consider as the objective function total duration plus lateness and assume that the optimal fleet size is given *a priori*. Haghani and Jung (2005) minimize the sum of costs associated with number of vehicles, distance, duration, and lateness. Fleischmann et al. (2004) minimize number of vehicles and total duration. Donati et al. (2008) optimizes fleet size (primary objective) and total route duration (secondary objective).

Summarizing, different solution approaches such as tabu search, ant colony, and genetic algorithms have been proposed for the VRPTD. The computational complexity of the existing TDVRP approaches have never been discussed or analyzed though some solution approaches are not fast as later discussed in Section 7 where solution quality and running times are compared. In addition, benchmark instances have not been yet proposed. Section 8 discusses the Iterative Route Construction and Improvement (IRCI) TDVRP algorithmic complexity; benchmark instances that can be clearly and unmistakably replicated by future researchers are detailed in Section 6. The next section introduces mathematical notation and defines the problem under study. To the best of the author's knowledge there is no published work that presents replicable time dependent vehicle routing problems with hard time windows constraints.

3. Problem definition

Using a traditional flow-arc formulation (Desrochers et al., 1988), the time dependent vehicle routing problem with hard time windows studied in this research can be described as follows. Let $G = (V, A)$ be a graph where $A = \{(v_i, v_j): i \neq j \wedge i, j \in V\}$ is an arc set and the vertex set is $V = (v_0, \ldots, v_{n+1})$. Vertices v_0 and v_{n+1} denote the depot at which vehicles of capacity $q_{\max}$ are based. Each vertex in V has an associated demand $q_i \geqslant 0$, a service time $g_i \geqslant 0$, and a service time window $[e_i, l_i]$; in particular the depot has $g_0 = 0$ and $q_0 = 0$. The set of vertices $C = \{v_1, \ldots, v_n\}$ specifies a set of n customers. The arrival time of a vehicle at customer i, $i \in C$ is denoted a_i and its departure time b_i. Each arc (v_i, v_j) has an associated constant distance $d_{ij} \geqslant 0$ and a travel time $t_{ij}(b_i) \geqslant 0$ which is a function of the departure time from customer i. The set of available vehicles is denoted K. The cost per unit of route duration is denoted c_t; the cost per unit distance traveled is denoted c_d. It is assumed that the problem is feasible, i.e. it is always possible to feasibly serve any individual customer starting from the depot.

The primary objective function for the TDVRP is the minimization of the number of routes; the optimal number of routes is unknown. A secondary objective is the minimization of total time or distance. There are two decision variables in this formulation; x_{ij}^k is a binary decision variable that indicates whether vehicle k travels between customers i and j. The real number decision variable y_i^k indicates service start time for customer i served by vehicle k. The TDVRP is formulated as follows:

Primary objective:

$$\text{minimize} \quad \sum_{k\in K}\sum_{j\in C} x_{0j}^k, \tag{1}$$

Secondary objective:

$$\text{minimize} \quad c_d \sum_{k\in K}\sum_{(i,j)\in A} d_{ij}^k x_{ij}^k + c_t \sum_{k\in K}\sum_{j\in C} (y_{n+1}^k - y_0^k) x_{0j}^k, \tag{2}$$

subject to:

$$\sum_{i\in C} q_i \sum_{j\in V} x_{ij}^k \leqslant q_{\max}, \quad \forall k \in K \tag{3}$$

$$\sum_{k\in K}\sum_{j\in V} x_{ij}^k = 1, \quad \forall i \in C \tag{4}$$

$$\sum_{i\in V} x_{il}^k - \sum_{i\in V} x_{ij}^k = 0, \quad \forall l \in C, \ \forall k \in K \tag{5}$$

$$x_{i0}^k = 0, x_{n+1,i}^k = 0, \quad \forall i \in V, \ \forall k \in K \tag{6}$$

$$\sum_{j\in V} x_{0j}^k = 1, \quad \forall k \in K \tag{7}$$

$$\sum_{j\in V} x_{j,n+1}^k = 1, \quad \forall k \in K \tag{8}$$

$$e_i \sum_{j\in V} x_{ij}^k \leqslant y_i^k, \quad \forall i \in V, \ \forall k \in K \tag{9}$$

$$l_i \sum_{j \in V} x_{ij}^k \geqslant y_i^k, \quad \forall i \in V, \ \forall k \in K \tag{10}$$

$$x_{i,j}^k (y_i^k + g_i + t_{i,j}(y_i^k + g_i)) \leqslant y_j^k, \quad \forall (i,j) \in A, \ \forall k \in K \tag{11}$$

$$x_{ij}^k \in \{0,1\}, \quad \forall (i,j) \in A, \ \forall k \in K \tag{12}$$

$$y_i^k \in \Re, \quad \forall i \in V, \ \forall k \in K \tag{13}$$

The primary and secondary objectives are defined by (1) and (2) respectively. The constraints are defined as follows: vehicle capacity cannot be exceeded (3); all customers must be served (4); if a vehicle arrives at a customer it must also depart from that customer (5); routes must start and end at the depot (6); each vehicle leaves from and returns to the depot exactly once, (7) and (8) respectively; service times must satisfy time window start (9) and ending (10) times; and service start time must allow for travel time between customers (11). Decision variables type and domain are indicated in (12) and (13).

In the TDVRP with soft time windows, customer service time windows are defined by two intervals $[e_i, l_i]$ and $[e_i^\#, l_i^\#]$ where $e_i \leqslant e_i^\#, l_i^\# \leqslant l_i$. The interval $[e_i^\#, l_i^\#]$ indicates the interval of time where service can start without incurring a penalty. The interval $[e_i, l_i]$ indicates the interval of time where service can start but there are additional costs, associated to c_e or c_l, if service starts early or late, respectively – i.e. during the early interval $[e_i, e_i^\#]$ or during the late interval $[l_i^\#, l_i]$. Defining x_i^e and x_i^l as auxiliary binary variables that indicate whether a penalty is incurred, the objective functions can be expressed as follows:

$$\text{minimize} \quad \sum_{i \in C} x_i^l, \quad \forall i \in C \tag{14}$$

$$\text{minimize} \quad \sum_{i \in C} x_i^e, \quad \forall i \in C \tag{15}$$

$$\text{minimize} \quad c_d \sum_{k \in K} \sum_{(i,j) \in A} d_{ij}^k x_{ij}^k + c_t \sum_{k \in K} \sum_{j \in C} (y_{n+1}^k - y_0^k) x_{0j}^k + c_e \sum_{k \in K} \sum_{i \in C} (e_i^\# - y_i^k)^+ + c_l \sum_{k \in K} \sum_{i \in C} (y_i^k - e_l^\#)^+ \tag{16}$$

subject to

$$x_i^l (y_i^k - e_l^\#)^+ \leqslant x_i^l \tag{17}$$

$$x_i^e (e_i^\# - y_i^k)^+ \leqslant x_i^e \tag{18}$$

$$x_i^e \in \{0,1\}, \quad x_i^l \in \{0,1\} \tag{19}$$

The primary objective function for the TDVRP with soft time windows is still the minimization of the number of routes. Using a customer service perspective ranking, a secondary objective is the minimization of the number of late penalties[1] (14); a tertiary objective is the minimization of early penalties (15); a final objective is the minimization of the combined distance, route duration, and soft time window costs (16). Logical constraints (17) and (18) are used to determine if service times must be penalized due to early or late time window utilization, respectively.

It is important to notice that the depot time windows as well as the maximum route duration are not changed as a result of the customers' time window relaxation. The TDVRP with hard time windows is a special case of the soft time window formulation. If $e_i = e_i^\#$ and $l_i^\# = l_i$, then (14) and (15) are redundant and (16) is reduced to (2). The travel time speed in any arc is a positive and continuous function of time, $s_{ij}(t) > 0$, which guarantees the FIFO property (Ahn and Shin, 1991). In addition, in the presented TDVRP travel times may be asymmetrical, i.e. $t_{i,j}(y_i^k) \neq t_{j,i}(y_j^k)$ even if $y_i^k = y_j^k$.

Unlike previous formulations of the TDVRP (Malandraki, 1989; Jung and Haghani, 2001) time is not partitioned into discrete intervals. Furthermore, the decision variable y_i^k allows for waiting at customer i; service start time may not necessarily be the same as arrival time. For example, if the vehicle arrives too early, it can wait at the customer location to avoid early service penalties. However, waiting may have an impact on future travel times. The following two sections describe a solution approach to tackle the TDVRP.

4. Solution approach

Time dependent travel times require significant modifications to local search approaches and metaheuristics that have been successfully applied to the traditional constant time VRPTW (Braysy and Gendreau, 2005a,b). A customer insertion or a local improvement not only influences the arrival and departure times of a "local" subset of customers but it may also

[1] Although the cost of early and late service times are application dependent, in numerous real life problems early services are preferred over late services, e.g. blood transport, just-in-time production systems, express mail delivery, etc.

significantly change travel times among "local" customers. Furthermore, the impact of altering a routing sequence is not just "local" but potentially affects all subsequent travel times. Changes in travel times have a subsequent impact on feasibility. To a certain degree, introducing soft time windows ameliorates the computational burden and loss of efficiency introduced by time dependent travel times. However, hard time constraints are more difficult to accommodate and this is reflected in the literature review. There are no published results in a set of standard benchmark problems with hard time windows and time dependent travel times. The absence of standard problems can also be attributed to the multiple different versions of the time dependent VRP.

The presented Iterative Route Construction and Improvement (IRCI) solution approach for the TDVRP employs algorithms that do not require modifications to accommodate constant or time dependent travel speeds. This research builds upon previous work to solve the constant speed VRP with soft and hard time windows (Figliozzi, 2008, 2010b).

The solution method to minimize fleet size is divided into two phases and algorithms: route construction and route improvement. A third algorithm, an auxiliary route building heuristic, is reiterated during the execution of the construction heuristic. The construction and auxiliary algorithms are *sequential*, i.e. the routes are built sequentially one customer at the time but always looking ahead at the cost of building a feasible solution when a given customer is routed or sequenced. Complete and feasible solutions are always returned by the route construction algorithm. The following pseudo-code illustrates the interaction between the route construction algorithm and the auxiliary routing algorithm in the construction phase (the calls are underlined). Sets are denoted by capital letters in *italics* (for example set of customers C); functions are denoted by small letters in bold (for example the cost function $\mathbf{c}(C)$); sub-algorithms are denoted by bold capital letters (for example $\mathbf{H}_r$).

4.1. Construction phase

Start
1 Read problem data
2 Input initial search space $\mathbf{\Delta}$
3 **for** each $\Delta \in \mathbf{\Delta}$
4 Call route construction algorithm for the set of unrouted customers C
5 Set depot as the initial customer i
6 **do**
7 Select a set of customer candidates J to follow i, $J \in C$
8 Call the auxiliary routing algorithm for each pair (i, j) where $j \in J$
9 Find pair $(i, j)^*$ with least cost solution and route i
10 Update best solution Z^* if there was an improvement
11 Update initial customer, $i \leftarrow j$, remove i from C
12 **while** $J \neq \{\}$
13 Return best solution found Z^* and corresponding Δ
14 **end for**
Output:
Best solution found $Z^*$and corresponding Δ

The second pseudo-code illustrates the interaction between the route improvement algorithm and the route construction algorithm. The route construction algorithm is executed (one or more times) every time the improvement algorithm is executed. The improvement procedure is also at the *route* level, i.e. it is not a local improvement since improvements are obtained by grouping routes and improving a set of routes at the time.

4.2. Improvement phase

Start
1 Input best solution found Z^* and corresponding Δ
2 Input criteria to select subsets of routes z^* from Z^*, $z^* \subset Z^*$
3 Form set Ω of subsets z^* to explore
4 **For** each subset $z^* \in \Omega$
5 Call route construction algorithm for customers in z^*
6 Return best solution found z'
7 If z' improves the partial solution z^*, then replace z^* and update Z^*
8 **end for**
Output:
Best solution found Z^*

The same basic algorithm applies to problems with soft and hard time windows. Hence, the presented algorithm produces solutions for time dependent vehicle routing problems with hard and soft time windows with similar computation times.

Using a bottom up approach the detailed descriptions of the algorithms are introduced in the following order: (a) an algorithm to sequence any given set of customers (auxiliary algorithm), (b) the route construction algorithm, and (c) the route improvement algorithm. In addition, due to the nature of the TDVRP, advancing or delaying service time may have a favorable impact on future travel times and costs. Hence another algorithm (d) is described in Section 5 to optimize service times in problems with soft time windows given a set of routes obtained from (c). Travel time calculations are necessary to execute (a)–(d). However, unlike algorithms (a)–(d), travel time calculations are heavily dependent on the specific type of speed function. Hence, the algorithm used to calculate travel times is presented in Appendix A for a specific type of speed function.

4.3. The auxiliary routing algorithm $\mathbf{H}_r$

The auxiliary routing algorithm $\mathbf{H}_r$ can be any heuristic that given a starting vertex, a set of customers, and a depot location returns a set of *routes* that satisfy the constraints of the TDVRP with soft or hard time windows. The auxiliary route heuristic is defined as $\mathbf{H}_r(\Delta, v_i, C, v_0)$ where $\Delta = \{\delta_0, \delta_1, \ldots, \delta_6\}$ are the parameters of the generalized cost function, v_i is the vertex where the first route starts, C is the set of customers to route, and v_0 is the depot where all routes end and all additional routes start, with the exception of the first route which starts at v_i.

In this research, $\mathbf{H}_r$ is a generalized nearest-neighbor heuristic (GNNH). The GNNH starts every route k by finding, from a subset of C, the unrouted customer with the least appending "generalized cost". At every subsequent iteration, the heuristic searches for the remaining unrouted customer with the least appending cost. Let i denote the initial vertex and let j denote a potential customer to append next. Let q_i^k denote the remaining capacity of the vehicle k after serving customer i. The service at customer i in route k begins at the earliest feasible time, which is $y_i^k = \max(a_i, e_i)$, and the departure time is given by $y_i^k + g_i$. The generalized cost of going from customer i to customer j is estimated as:

$$\mathbf{g}(\Delta, i, j, k) = \delta_1 d_{ij} + \delta_2(y_j^k - (y_i^k + g_i)) + \delta_3(l_j - (y_i^k + g_i + t_{ij}(y_i^k + g_i))) + \delta_4(q_i^k - q_j) + \delta_5[e_j^{\#} - y_j^k]^+ + \delta_6[y_j^k - e_j^{\#}]^+ \quad (20)$$

If customer j is infeasible, i.e. it cannot be visited after serving customer i, the cost of ending customer i's route and starting a new one to serve customer j is estimated as:

$$\mathbf{g}(\Delta, i, j) = \delta_0 + \delta_1 d_{0j} + \delta_2(y_j^k - e_0) + \delta_3(l_j - t_{0j}(e_0)) + \delta_4(q_{\max} - d_j) + \delta_5[e_j^{\#} - y_j^k]^+ + \delta_6[y_j^k - e_j^{\#}]^+ \quad (21)$$

where δ_0 is the cost of adding a new vehicle.

The parameter δ_1 takes into account the relative distance between customers. The parameter δ_1 is included, even in problems with time windows, to reduce route length; this is particularly important in instances where time windows are wide or not "binding" in all routes. The parameter δ_2 accounts for the "slack" between the completion of service at i and beginning of service at j. Following Solomon's approach (Solomon, 1987), the parameter δ_3 takes into account the "urgency" of serving customer j, expressed as the time remaining until the vehicle's last possible service time start. The parameter δ_4 takes into account the capacity slack of the vehicle after serving customer j. The parameters δ_5 and δ_6 are added to account for possible early or late service penalties, respectively.

4.4. The route construction algorithm $\mathbf{H}_c$

In this algorithm, denoted $\mathbf{H}_c$, routes are constructed sequentially. Given a partial solution and a set of unrouted customers, the algorithm uses the auxiliary heuristic $\mathbf{H}_r$ to search for the feasible least cost set of routes. The algorithm also uses an auxiliary function $\mathbf{w}(v_i, C, g, W)$ that, given a set of unrouted customers C, a vertex v_i, $v_i \notin C$, and a generalized cost function $\mathbf{g}(\Delta, i, j, k)$, returns a set of vertices of cardinality W with the lowest generalized costs $\mathbf{g}(\Delta, i, j, k)$ for all $v_j \in C$. The function $\mathbf{c}(sequenceofcustomers)$ simply evaluates the *true* cost of a sequence of costumers utilizing the objective function (16).

Parameters:

$\mathbf{H}_r$:	Route building heuristic
W:	Width of the search, the number of solutions to be built and compared before adding a customer to a route, $W \leqslant \lvert C \rvert$
$\mathbf{\Delta}$:	Search space of the route heuristic generalized cost parameters, each $\Delta \in \mathbf{\Delta}$ is a vector of cost parameters

Data:

C:	Set of customers to route
v_0:	the depot
v_i:	initial vertex (customer that is last in the partially formed route, or the depot in the special case when the route is empty)

(con

Start $\mathbf{H}_c$
1 $start \leftarrow v_0$
2 $lowestCost \leftarrow \infty$
3 $bestSequence \leftarrow v_0$
4 **for** each $\Delta \in \mathbf{\Delta}$
5 **while** $C \neq \varnothing$ **do**
6 $C^* \leftarrow w(start, C, g, W)$
7 **for** each $v_i \in C^*$
8 **if** $\mathbf{c}(bestSequence \cup \mathbf{H}_r(\Delta, v_i, C, v_0)) < lowestCost$ **then**
9 $lowestCost \leftarrow \mathbf{c}(bestSequence \cup \mathbf{H}_r(\Delta, v_i, C, v_0))$
10 $lowestNext \leftarrow v_i$
11 **end if**
12 **end for**
13 $start \leftarrow lowestNext$
14 $C \leftarrow C \setminus lowestNext$
15 $bestSequence \leftarrow bestSequence \cup lowestNext$
16 $R \leftarrow bestSequence \cup \mathbf{H}_r(\Delta, lowestNext, C, v_0)$
17 **end while**
18 **end for**
Output:
Best set of routes found R to serve all C customers (R and its associated Δ)
END $\mathbf{H}_c$

This algorithm will sequentially construct routes. In line 6, up to W "potentially" good candidates are selected. Line 8 attempts to "look ahead" and estimate the future costs of adding v_i to the existing sequence of routed customers. Between lines 7 to 12 complete routes (built using the W candidates and the heuristic $\mathbf{H}_r$) are compared. The candidate with the least generalized cost is selected. The generalized cost function g that is used in $\mathbf{H}_r$ must not be confused with the objective cost function c that is used in $\mathbf{H}_c$ or the improvement heuristic $\mathbf{H}_i$; the latter cost function is the sum of the accrued vehicle, distance, time, or penalty costs as indicated in the objective function.

Each $\Delta \in \mathbf{\Delta}$ is likely to produce a different solution in terms of fleet size, distance traveled, and route duration. The search space is limited by these constraints:

$$\delta_1 + \delta_2 + \delta_3 = 1$$

The parameter δ_4 is bounded by zero and the ratio between the median intercustomer distance and the median customer demand. The parameters δ_5 and δ_6 are bounded between zero and the possible early or late service penalties, respectively. All parameters are assumed to be non-negative.

4.5. The route improvement algorithm $\mathbf{H}_i$

The route construction algorithm generates an initial set of routes. Any route obtained from $\mathbf{H}_c$ is a special cluster of customers with the desirable property that there is at least one feasible sequence that satisfies all the constraints of the TDVRP. Fleet size and routing costs can be further reduced using a *route improvement* algorithm. The improvement algorithm works on a subset of routes. The gist of the algorithm resembles the ruin and recreate approach presented by Schrimpf et al. (2000) for the TSP and VRP.

The motivation of the route improvement algorithm is to combine these routes, "feasible clusters", to consolidate or improve the efficiency of routes that are not fully utilized in terms of vehicle capacity, route duration, number of customers serviced, etc. In the $\mathbf{H}_i$ algorithm two functions are introduced. The function $\mathbf{k}_s(R, s)$ orders the set of routes R from smallest to largest based on the number of customers per route and then returns a set of $s \geqslant 1$ routes with the least number of customers; e.g. $\mathbf{k}_s(R, 1)$ will return the route with the least number of customers. The number of customers per route is used as a proxy measure of potential route utilization in terms of duration or vehicle capacity. If two or more routes have the same number of customers, ties are solved drawing random numbers.

The function $\mathbf{k}_g(r_i, S, p)$ returns a set of p routes that belong to S (a complete solution already) and are good matches for route r_i. The term "good matches" refers to routes that as a group have the potential to be consolidated or improved. In this research a linear combination of two distinct measures are used to evaluate the quality of a potential "match": (1) geographical proximity – the distance between any two routes' center of gravity is used as a proxy measure of geographic proximity assuming that close routes have the potential to be improved and (2) utilization – the number of customers per route is used as a proxy measure of potential route capacity utilization assuming that routes are poorly utilized have the potential to be combined. Other measures of geo-temporal proximity or utilization can be used to find good potential matches. By definition, the route r_i is always included in the output of the set function $\mathbf{k}_g(r_i, S, p)$. To simplify notation the term $C(G)$ is the set of customers served by the set of routes G.

Data:
R: Set of routes
Parameters:
$\mathbf{H}_c$: Route construction heuristic
W: Number of solutions to be built and compared in the construction heuristic
Δ: Generalized cost parameters of the auxiliary routing algorithm
s: Number of routes potentially considered for improvement
p: Number of neighboring routes that are reconstructed
$\mathbf{k}_s$ and $\mathbf{k}_g$: Route selection functions

Start $\mathbf{H}_i$
1 $s \leftarrow \min(s, |R|)$
2 $p \leftarrow \min(s, p)$
3 $S \leftarrow \mathbf{k}_s(R, s) \subseteq R$
4 $S' \leftarrow R \setminus S$
5 $r^* \leftarrow \mathbf{k}_s(S, 1)$
6 **while** $|S| > 1$ **do**
7 $G \leftarrow \mathbf{k}_g(r^*, S, p)$
8 $G' \leftarrow \mathbf{H}_c(\mathbf{H}_r, W, \Delta, C(G))$
9 **if** $\mathbf{c}(G') < \mathbf{c}(G)$ **then**
10 $R \leftarrow R \setminus G$
11 $R \leftarrow R \cup G'$
12 $S \leftarrow S \setminus G$
13 $S \leftarrow S \cup G'$
14 $r^* \leftarrow \mathbf{k}_s(S, 1)$
15 **end if**
16 $r \leftarrow S \setminus \mathbf{k}_s(S, |S| - 1)$
17 $S \leftarrow S \setminus r$
18 **if** $|S'| > 0$ **then**
19 $r' = \mathbf{k}_s(S', 1)$
20 $S' \leftarrow S'/r'$
21 $S \leftarrow S \cup r'$
22 **end while**
Output:
R set of improved routes
END $\mathbf{H}_i$

The smallest route is selected in line 5 using function $\mathbf{k}_s$ while "matching" routes are selected in line 7 using function $\mathbf{k}_g$. The routes are reconstructed in line 8. If the generalized cost is improved (line 9), the routes with the least generalized cost are selected. The algorithm continues adding new routes until all routes have had at least one opportunity to be reconstructed and "re-optimized".

5. Service time improvement

The previous algorithms deal with the minimization of costs via sequencing of customers and their assignment to routes. The $\mathbf{H}_{yf}$ and $\mathbf{H}_{yb}$ algorithms aim to reduce costs, if total route duration is the key secondary objective, by improving customer service start times for a given set of routes produced by $\mathbf{H}_i$. This goal has already been analyzed by Desrosiers et al. (1995) in the context of scheduling traditional VRP fixed routes.

For any given route k, an approach similar to dynamic programming can be used to determine the optimal service start times y_i^k for customer i belonging to route k given the arrival time a_i; each customer is associated with a stage, the decision variable is the service time y_i^k, and the state is defined by the arrival time a_i. For the sake of notational simplicity, let's denote a sequence of vertices in a route by their location in the route; hence, any given route k is defined by the sequence of customers $(0, 1, 2, \ldots, q, q+1)$ where 0 and $q+1$ denote the depot. If the cost to minimize is the sum of distance traveled, route durations, and soft time window utilization given by Expression (16), the cost function, $\pi(y_q, a_q)$, for the last customer is[2]:

$$\pi(y_q, a_q) = c_d d_{q,q+1} + c_t(a_{q+1} - y_q) + c_e(e_q^{\#} - y_q^k)^+ + c_l(y_q^k - e_l^{\#})^+ \tag{22}$$

where $a_{q+1} = y_q + g_q + t_{q,q+1}(y_q + g_q)$ and subject to $l_q \geqslant y_q \geqslant a_q$.

[2] The distance term can be eliminated from (22) because it is not affected by service time.

Using a backward solution approach, for each customer it is possible to define a stage cost and an optimal cost to go function. Further, for each customer, it is possible to limit the feasible space of customer service time to a closed time interval. For a customer i belonging to route k, let $\underline{y}_i^k$ and $\bar{y}_i^k$ denote, respectively, the earliest and latest feasible service times.

Lemma 1. *Given* any route k, the optimal service times at any customer i belong to the time interval $[\underline{y}_i^k, \bar{y}_i^k]$ and can be calculated using a forward and backward algorithm.

Proof 1. Starting from the depot, earliest possible arrival at customer 1 is $a_1 = e_0 + t_{01}(e_0)$ due to FIFO property; earliest service time at customer 1 is $\underline{y}_1^k = \max(a_1, e_1)$; earliest departure at customer 1 is $\underline{y}_1^k + g_1$. Earliest possible arrival at customer 2 is $a_2 \underline{y}_1^k + g_1 + t_{1,2}(\underline{y}_1^k + g_1)$ due to FIFO property; earliest service time at customer 2 is $\underline{y}_2^k = \max(a_2, e_2)$; earliest departure at customer 2 is $\underline{y}_2^k + g_2$ and so on until reaching the last customer.On the other hand, starting from the depot, latest possible departure time last from customer q is: $\arg\max_{y \in \Re} y, s.t.(y + t_{q,q+1}(y) \leqslant l_{q+1})$; due to the continuous speed function and the FIFO property this value is unique. The latest possible service time at customer q is $\bar{y}_q^k = \min(y - g_q, l_q)$. Latest possible departure time from customer $q-1$ is $\arg\max_{y \in \Re} y, s.t.(y + t_{q-1,q}(y) \leqslant \bar{y}_q^k)$; latest possible service time at customer $q-1$ is $\bar{y}_{q-1}^k = \min(y - g_{q-1}, l_{q-1})$ and so on until reaching the first customer. □

It is possible to state properties that simplify the determination of service start times.

Property 1. *Given a route k outputted by* $\mathbf{H}_i$*, the customer service times are the earliest feasible times.*

Proof. Due to the workings of the $\mathbf{H}_r$ algorithm, the service at any customer i in route k begins at the earliest feasible time, which is $y_i^k = \underline{y}_i^k = \max(a_i, e_i)$, and the departure time is given by $y_i^k + g_i$. Due to the FIFO property, for the given routes, customers cannot be serviced earlier than the provided service start times. □

Property 2. *Given a route k outputted by* $\mathbf{H}_i$*, total route duration cannot be reduced further.*

Proof. Due to Property 1, service times cannot be advanced. Then, the FIFO property guarantees that route duration cannot be reduced further unless the set of routes is altered. The arrival times at each customer are the earliest possible for the sequence given by route k. □

Property 3. *Given a route k outputted by* $\mathbf{H}_i$*, a TDVRP with hard time windows requires no service time optimization for route k.*

Proof. Due to Property 2, route durations cannot be reduced. Start times do not affect distance traveled and there are no soft time windows penalties or costs to be reduced. Hence, altering service start time will not reduce any objective function. □

Property 4. *Given a route k outputted by* $\mathbf{H}_i$*, if a customer uses the "late" soft time window, no improvement can be made by changing the service time.*

Proof. Due to Property 1, the service time cannot be advanced without losing feasibility. If the service time is delayed, there is a greater late penalty. Hence, if a customer in the route outputted by $\mathbf{H}_i$ uses a late time window, the provided service time for that customer cannot be improved. □

Corollary. *In a route outputted by* $\mathbf{H}_i$*, the service time optimization problem can be decomposed into smaller problems delimited by customers using "late" soft time windows.*

5.1. Service time improvement algorithms

For each customer that uses an early soft time window, the $\mathbf{H}_{yb}$ algorithm attempts to reduce early soft time window usage without allowing the introduction of service delays that increase late time window usage. This algorithm operates backwards. For the sake of presentation simplicity, periods of constant travel time are assumed. The depot working time $[e_0, l_0]$ is partitioned into p time periods $\mathbf{T} = T_1, T_2, \ldots, T_p$; each period T_k has an associated constant travel speed s_k in the time interval $T_k = [t_{\underline{k}}, t_{\bar{k}}]$.

Data:
T and S: Time intervals and speeds
v_i, v_j, y_j^k: Two customers served in this order in route k, y_j^k is the current service time at customer j
START $\mathbf{H}_{yb}$
1 **If** $y_j^k < l_j^{\#}$ & $y_j^k < \bar{y}_j^k$ **then**
2 $y_j^k \leftarrow \min(l_j^{\#}, \bar{y}_j^k)$
3 **end if**
4 **find** k, $t_{\underline{k}} \leqslant y_j^k \leqslant t_{\bar{k}}$
5 $b_i \leftarrow y_j^k - d_{ji}/s_k$
6 $d \leftarrow d_{ji}, t \leftarrow y_j^k$
7 **while** $b_i < t_{\underline{k}}$ **do**
8 $d \leftarrow d - (t - t_{\underline{k}})S_k$
9 $t \leftarrow t_{\underline{k}}$
10 $b_i \leftarrow t - d/s_{k+1}$
11 $k \leftarrow k + 1$
12 **end while**
13 $\bar{y}_i^k \leftarrow \min(b_i - g_i, l_i)$
Output:
$y_j^k, \bar{y}_i^k$
END $\mathbf{H}_{yb}$

After early time windows have been reduced, a final task is to reduce route duration without increasing the number of late soft time windows. The following forward algorithm, $\mathbf{H}_{yf}$, reduces route duration without increasing soft time windows.

Data:
T and S: Time intervals and speeds
v_i, v_j, y_j: Two customers served in this order in route k, y_i^k is the current service time at customer j
START $\mathbf{H}_{yf}$
1 **if** $y_i^k > e_j^{\#}$ & $y_i^k > \underline{y}_i^k$ **then**
2 $y_i^k \leftarrow \max(e_i^{\#}, \underline{y}_i^k)$
3 **end if**
4 **find** $k, t_{\underline{k}} \leqslant y_i^k \leqslant t_{\bar{k}}$
5 $a_j \leftarrow y_i^k + d_{ij}/s_k$
6 $d \leftarrow d_{ij}, t \leftarrow y_i^k$
7 **while** $a_j > t_{\bar{k}}$ **do**
8 $d \leftarrow d - (t_{\bar{k}} - t)s_k$
9 $t \leftarrow t_{\underline{k}}$
10 $a_j \leftarrow t + d/s_{k+1}$
11 $k \leftarrow k + 1$
12 **end while**
13 $\underline{y}_j^k \leftarrow \max(a_j, e_j)$
Output:
$y_i^k, \underline{y}_j^k$
END $\mathbf{H}_{yf}$

Both algorithms try to reduce the interval $[\underline{y}_i^k, \bar{y}_i^k]$ where the optimal service start time is found for a given route k.

6. Proposed benchmark problems

Barr et al. (1995) presented a detailed discussion of the how to properly test heuristics methods and provide useful testing result. These authors emphasized *reproducibility* as an "...essential ingredient of scientific research" further indicating that "experimental results that cannot be independently verified are given little credence in the scientific community". Although this may seem too obvious for many researchers in the physical sciences, it is unfortunately not followed by many papers dealing with vehicle routing problems. Clear instructions regarding computing environment, testing, quality of the solution, parameter selection, statistics, variability, analysis and interpretation, and reporting were provided by the authors.

A few years later, Golden et al. (1998) expanded on some of the key problems and emphasized the need for *parsimonious, efficient, robust and simple* algorithms indicating that "In fact, some current heuristics suffer from inelegant design: they have more parameters than the number of benchmark problems! Furthermore, we suspect that heuristics are being over fit on the benchmark problems. In the future, the goal should be to design VRP heuristics that are capable of producing high-quality nearly optimal solutions. Heuristics should be lean and parsimonious and contain few parameters. They should be computationally efficient, robust, and simple". Unfortunately, to the best knowledge of the author, there has not been a change in the way results are presented and algorithms are tested; simplicity, robustness, and parsimoniousness are not properly discussed or analyzed in VRP related papers.

As later also indicated by Cordeau et al. (2002), results presented in the VRP literature usually present better results on benchmark problems at the expense of (a) too many parameters or complicated coding that lacks flexibility to accommodate real-life constraints, (b) too many parameters that are difficult to calibrate or even understand, and (c) solution approaches that are markedly tailored to perform well on the benchmark problems but that may lack generality and robustness in real-life problems. Furthermore, for solutions approaches that are not deterministic because they employ randomization, e.g. tabu search or GRASP, the testing and presentation of the results must be more demanding and include the variability of the solution quality and statistical testing instead of simply reporting "best results" or "averages" after the algorithm is fine tuned (Taillard, 2001; Golden et al., 1998).

As mentioned in Section 2, results provided in previous research efforts cannot be compared in terms of solution quality or computational time. This is revealing of a still incipient body of work for the TDVRP. The proposed set of benchmark problems are based on the classical instances of the VRP with time windows proposed by Solomon (1987). The Solomon instances include distinct spatial customer distributions, vehicles' capacities, customer demands, and customer time windows. These problems have not only been widely studied in the operations research literature but the datasets are readily available.[3]

The well-known 56 Solomon benchmark problems for vehicle routing problems with hard time windows are based on six groups of problem instances with 100 customers. The six problem classes are named C1, C2, R1, R2, RC1, and RC2. Customer locations were randomly generated (problem sets R1 and R2), clustered (problem sets C1 and C2), or mixed with randomly generated and clustered customer locations (problem sets RC1 and RC2). Problem sets R1, C1, and RC1 have a shorter scheduling horizon, tighter time windows, and fewer customers per route than problem sets R2, C2, and RC2 respectively.

This section proposes new test problems that capture the typical speed variations of congested urban settings which generally include slower travel speeds during morning and evening rush hours. The problems are divided into three categories of study: (1) constant speed Solomon instances, (2) time dependent problems with hard time windows, and (3) time dependent problems with soft time windows. Some previous research efforts may have used standard problems but they allocated travel speed distributions *randomly* to customer arcs or it is ambiguous the type of time dependency allocated to each arc. In order to provide readily replicable instances, the travel speed distributions apply to *ALL* arcs among customers, i.e. in the arc set:

$$A = \{(v_i, v_j) : i \neq j \wedge i, j \in V\}.$$

Most recent research efforts, as stated in Section 2, have used constant speed intervals. The same approach is adopted in this research because constant speed intervals guarantee the FIFO property and can be readily replicated. The algorithm used to calculate travel times is presented in Appendix A.

6.1. Constant speed problems with hard time windows

Constant travel speed is a special case of the general time dependent problem. These instances are the classical Solomon problems that have been widely studied and provide an indication of the performance of the algorithm with constant travel speed.

6.2. Time dependent problems with hard time windows

There could be many potential speed distributions. The goal is to present meaningful speed distributions while minimizing as much as possible the number of speed distributions and test problems. Solomon's instances represent the archetypical and ubiquitous case where a central depot services a set of surrounding customers with delivery time windows during the depot open time. Assuming a typical urban area with morning and evening congested periods and also assuming that the depot open time represents the duration of the vehicles/drivers workday, four distinct cases are proposed and interpreted as follows:

(a) The trucks leave the depot very early in the day and can travel without congestion during the first 1/5 of the workday before morning congestion takes place. The following 1/5 of the workday coincides with morning congestion followed by the somewhat less congested midday period. Another 1/5 of the workday coincides with evening congestion and the last 1/5 of the workday takes place after congestion has subsided.

[3] Several websites maintain downloadable datasets of the instances including Solomon's own website: http://web.cba.neu.edu/~msolomon/problems.htm.

(b) The opposite of the case (a); the trucks leave the depot just as morning congestion starts for 1/5 of the workday, the following 1/5 of the workday coincides with no congestion followed by the somewhat congested midday period. Another 1/5 of the workday takes place before evening congestion and the last 1/5 of the workday coincides with the heavy evening congestion.
(c) The trucks leave the depot early in the day so that the first 1/2 of the workday is uncongested followed by the second 1/2 workday that takes place during the heaviest morning congested period.
(d) Due to time window constraints, the trucks leave the depot just as morning congestion begins; hence, the initial 1/2 workday takes place under heavy morning congestion and the second 1/2 of the workday takes place after congestion has subsided.

Because the Solomon problems are tightly constrained, i.e. a small decrease in travel speed results in infeasible problems due to the hard time windows, the only approach to leverage the existing Solomon instances to evaluate time dependent problems is to introduce speed variability with higher travel speeds (the speed in the original Solomon problems is constant and equal to one). The maximum ratio in travel speeds is 2.5–1. This ratio is chosen because it is realistic to assume that in an urban freeway the free-flow or uncongested travel speed is around 60 miles per hour and with heavy congestion the speed drops to 24 miles per hour. A time dependent problem is generated by assigning the speed distribution to all arcs to facilitate reproducibility.

Type (a)

These instances introduce fast periods between depot opening and closing times. The depot working time $[e_0, l_0]$ is divided into five time periods of equal durations:

- $[0, 0.2l_0)$; $[0.2l_0, 0.4l_0)$; $[0.4l_0, 0.6l_0)$; $[0.6l_0, 0.8l_0)$; and $[0.8l_0, l_0]$.

and the corresponding travel speeds are:

- TD1a = [1.00, 1.60, 1.05, 1.60, 1.00],
- TD2a = [1.00, 2.00, 1.50, 2.00, 1.00],
- TD3a = [1.00, 2.50, 1.75, 2.50, 1.00].

Type (b)

The opposite to type (a), higher travel speeds are found at the extremes of the working day.

- TD1b = [1.60, 1.00, 1.05, 1.00, 1.60],
- TD2b = [2.00, 1.00, 1.50, 1.00, 2.00],
- TD3b = [2.50, 1.00, 1.75, 1.00, 2.50].

Type (c)

Higher travel speeds are found at the beginning of the working day.

- TD1c = [1.60, 1.60, 1.05, 1.00, 1.00],
- TD2c = [2.00, 2.00, 1.50, 1.00, 1.00],
- TD3c = [2.50, 2.50, 1.75, 1.00, 1.00].

Type (d)

The opposite to type (b), higher travel speeds are found at the end of the working day.

- TD1d = [1.00, 1.00, 1.05, 1.60, 1.60],
- TD2d = [1.00, 1.00, 1.50, 2.00, 2.00],
- TD3d = [1.00, 1.00, 1.75, 2.50, 2.50].

In all cases, types (a)–(d), if the vehicles were to travel non-stop in the interval $[e_0, l_0]$ the vehicle would travel an extra 25%, 50%, and 75% more for speeds TD1, TD2, and TD3 respectively than in the original Solomon instances.

6.3. Time dependent problems with soft time windows

These instances introduce two congested periods between depot opening and closing times. The depot working time is divided into the same five periods and the corresponding travel speeds are:

- TD4 = [1.10, 0.85, 1.10, 0.85, 1.10],
- TD5 = [1.20, 0.80, 1.00, 0.80, 1.20],
- TD6 = [1.20, 0.70, 1.20, 0.70, 1.20].

If one vehicle were to travel non-stop in the interval $[e_0, l_0]$, this vehicle would travel the same distance as in the original Solomon instances but with increasing travel speed variability, i.e. same average speed but with increased variability. However, soft time windows are required because some Solomon problems would be infeasible otherwise (Ichoua et al., 2003; Donati et al., 2008). An allowable time window violation per customer equal to: $P_{\max} = 0.1(l_0 - e_0) = e_i^{\#} - e_i = l_i - l_i^{\#}$ is allowed. However, the depot working time $[e_0, l_0]$ is not relaxed. The penalty cost for an early or late delivery is one unit of cost per unit time which is the same value used in constant speed Solomon instances with soft time windows (Balakrishnan, 1993; Chiang and Russell, 2004).

6.4. Time windows elasticity

As stated previously, Barr et al. (1995) and Golden et al. (1998) indicated that it was necessary to test in such a way that "overfitting" is avoided. Algorithmic parameters are always fine tuned to the benchmark instances but researchers do not publish the time needed to properly fine tune the parameters (Golden et al., 1998).

To better understand the response of the algorithm by problem type, a small perturbation is introduced. Given the importance of time windows, a small perturbation is introduced as follows:

$$e_i \leftarrow e_i + uniform[-1.0, 1.0] * 0.02 * (l_0 - e_0), \quad \forall i \in C$$

The magnitude of the perturbation cannot exceed 2% of the depot working time and the perturbation is randomly applied to each customer. The expectation of the sum of the perturbations equals zero or "neutral" since the expectation of the perturbations is:

$$E(uniform[-1.0, 1.0]) = 0$$

However, the expectation of the sum of the absolute perturbations is $0.01 * (l_0 - e_0)$ or 1% of the depot working time; i.e. conditional on being positive (or negative), the perturbations are on average 1% (−1%) of the depot working time.

There are two reasons for choosing absolute perturbations that are 1% on average:

(a) conceptual: a elasticity is conceptually described in the economic literature as the % change in the dependent variable when the independent variables is changed 1%, hence, the results of the perturbation analysis can be interpreted as the *elasticity* of the solution quality to a 1% change in the window start time, and
(b) practical: some problems or instances are so tight that they become infeasible if time window perturbations are larger than 1% on average.

7. Experimental results

Proper benchmarking of algorithms, solution quality and computation times can be performed using standardized instances and computers. However, computation times can be difficult to compare if there are significant differences in computer processing power or equipment. Detailed information regarding computer equipment (brand, model, processor, RAM) can be used to estimate relative computer power using Dongarra (2007) and SPEC[4] results. All the results presented in this section were obtained with a laptop Dell Latitude D430, with an Intel Core CPU 1.2 GHz and 1.99 GB of RAM. Even after standardizing problems and equipment there may be differences in running time due to different compilers, programming language, or code efficiency and implementation.

Golden et al. (1998) indicates that algorithms should be compared not only by the number of parameters but also by how intuitive and reasonable these parameters are from a user's perspective. To avoid excessive "tailoring", *all* the results presented in this research use the *exact* same procedure and parameter values in *all* cases, i.e. the same code, with the parameters described in Section 4, and the same parameter values. The exact same parameters are used not only in different types of problems, e.g. soft vs. hard, but also in different types of instances, e.g. R1 and C2. Travel time calculations were performed using the algorithm presented in Appendix A. It is also assumed that the algorithm does not "know" anything regarding the type of problem or its characteristics, e.g. average number of customers per route, binding constraints, or lower bounds. This type of information can be exploited to reduce computational times, e.g. usage of lower bounds (Figliozzi, 2008), but if new parameters, steps, or lines of code are needed they have to be explicitly stated to provide a level playing field when it comes to comparisons among algorithms. The hard time window results provided in this paper are found using a search space bounded by:

$$0.00 \leqslant \delta_1 \leqslant 0.65, 0.25 \leqslant \delta_2 \leqslant 1.00, 0.00 \leqslant \delta_3 \leqslant 0.20, 0.00 \leqslant \delta_4 \leqslant 0.03$$

and $\delta_1 + \delta_2 + \delta_3 = 1$; δ_0 is a sufficiently large number (primary objective is to minimize fleet size).

[4] Comparison among computers can be found at http://www.specbench.org/.

7.1. Constant speed problems with soft time windows

The first set of results corresponds to the extensively studied Solomon instances with constant travel speeds; results are presented in Table 1. In these instances the primary objective is to minimize the number of vehicles and the secondary objective to minimize travel distance. The first row presents the combination of the absolute best solutions found to date which have been obtained by different researchers, algorithms, machines, and computational times (Donati et al., 2008; SINTEF, 2008).

The second row presents the results of Taillard et al. (1997) using the tabu search algorithm for soft time window problems and constant travel speed that was implemented by Ichoua et al. (2003). Taillard et al. reported better results but at the expense of significantly longer computational times. The results reported by Taillard et al. and Donati et al. are average results and computation times over independent runs.

The performance of the IRCI algorithm, in relation to other approaches that can solve problems with both soft and hard time windows have been used in time dependent problems, is somewhat comparable. The IRCI solutions have relatively low computational times – an average of 21.3 s for each 100 customer problem but comparisons in terms of speed with Taillard et al. (1997) are difficult. Computers and their architecture have evolved significantly in the last 10 years. However, the IRCI is faster than the method presented by Donati et al. (2008). In terms of solution quality, the IRCI is outperformed by the best local search approaches (Braysy and Gendreau, 2005b). The IRCI solutions are, on average, slightly less than 4% from the best results ever obtained for the Solomon instances with constant travel times. The IRCI can obtain slightly better performances, around 3%, in terms of number of vehicles with longer computational times or by tailoring some parameters to each problem type. However, to avoid any kind of "distortion", the same general code is utilized to obtain all the results presented in this section.

7.2. Time dependent problems with hard time windows

The second set of results corresponds to the Solomon instances with time dependent travel speeds and hard time windows. In these instances the primary objective is to minimize the number of vehicles, the secondary objective is to minimize time and distance traveled.

To the best of the author's knowledge this is the first reporting of Solomon instances with hard time windows and time dependent speeds; results are presented in Tables 2–5. As expected, with increased travel speeds, the number of vehicles is reduced significantly. However, there is relatively minimal change in the distance traveled. Time traveled decreases as aver-

Table 1
VRPTW results for classical Solomon instances – constant speed.

Method	R1	R2	C1	C2	RC1	RC2
Average number of vehicles by problem class						
(1) Best Ever (1987–...)	11.92	2.73	10.00	3.00	11.50	3.25
(2) Taillard et al. (1997)	12.64	3.00	10.00	3.00	12.08	3.38
(3) Donati et al. (2008)	12.61	3.09	10.00	3.00	12.04	3.38
(4) IRCI	12.58	3.00	10.00	3.00	12.12	3.38
Average distance						
(1) Best Ever (1987–...)	1210	952	828	590	1384	1119
(2) Taillard et al. (1997)	1220	1013	828	591	1381	1199
(2) Donati et al. (2008)	1199	967	828	590	1374	1156
(4) IRCI	1248	1124	841	626	1466	1308

Computation time for all 56 problems: (1) different authors, machines and computation times; (2) Sun Sparc 10, 261 min; (3) Pentium IV 2.66 GHz, 168 min; (4) Dell Latitude D430, 1.2 GHz, 19.0 min.

Table 2
VRPTW results – hard time windows – type (a).

Travel time distribution	R1	R2	C1	C2	RC1	RC2
Average number of vehicles by problem class						
(1) TD1	11.67	2.82	10.00	3.00	11.38	3.25
(2) TD2	10.75	2.55	10.00	3.00	10.50	2.88
(3) TD3	9.92	2.27	10.00	3.00	10.00	2.75
Average distance						
(1) TD1	1295	1216	879	657	1405	1444
(2) TD2	1258	1244	864	654	1395	1454
(3) TD3	1237	1269	880	697	1362	1434
Average travel time						
(1) TD1	1080	990	729	563	1164	1177
(2) TD2	897	861	644	495	989	993
(3) TD3	793	774	608	485	860	867

Computation time for all 56 problems: (1) TD1, 19.1 min; (2) TD2, 17.7 min; (3) TD3, 17.3 min – in all cases using Dell Latitude D430, 1.2 GHz.

Table 3
VRPTW results – hard time windows – type (b).

Travel time distribution	R1	R2	C1	C2	RC1	RC2
Average number of vehicles by problem class						
(1) TD1	12.42	3.00	10.00	3.00	12.13	3.38
(2) TD2	11.50	2.73	10.00	3.00	11.25	3.25
(3) TD3	11.42	2.73	10.00	3.00	11.00	3.00
Average distance						
(1) TD1	1289	1212	892	670	1454	1403
(2) TD2	1279	1218	883	667	1429	1433
(3) TD3	1265	1245	866	714	1442	1483
Average travel time						
(1) TD1	1064	1027	732	545	1180	1200
(2) TD2	905	893	650	467	1010	1053
(3) TD3	808	831	584	446	916	981

Computation time for all 56 problems: (1) TD1, 19.7 min; (2) TD2, 19.0 min; (3) TD3, 18.7 min – in all cases using Dell Latitude D430, 1.2 GHz.

Table 4
VRPTW results – hard time windows –type (c).

Travel time distribution	R1	R2	C1	C2	RC1	RC2
Average number of vehicles by problem class						
(1) TD4	11.67	2.73	10.00	3.00	11.50	3.25
(2) TD5	10.83	2.55	10.00	3.00	10.75	2.75
(3) TD6	10.17	2.36	10.00	3.00	10.13	2.75
Average distance						
(1) TD4	1302	1245	865	683	1435	1407
(2) TD5	1266	1238	863	658	1413	1472
(3) TD6	1272	1243	862	665	1409	1438
Average travel time						
(1) TD4	1066	1003	697	573	1186	1147
(2) TD5	881	843	618	483	1012	1027
(3) TD6	801	760	565	451	904	886

Computation time for all 56 problems: (1) TD4, 19.6 min; (2) TD5, 18.2 min; (3) TD6, 18.1 min – in all cases using Dell Latitude D430, 1.2 GHz.

Table 5
VRPTW results – hard time windows – type (d).

Travel time distribution	R1	R2	C1	C2	RC1	RC2
Average number of vehicles by problem class						
(1) TD4	12.25	3.00	10.00	3.00	12.00	3.38
(2) TD5	11.58	2.73	10.00	3.00	11.25	3.25
(3) TD6	11.08	2.64	10.00	3.00	10.75	3.25
Average distance						
(1) TD4	1311	1218	872	666	1425	1394
(2) TD5	1272	1216	856	679	1404	1412
(3) TD6	1293	1215	867	690	1436	1424
Average travel time						
(1) TD4	1114	1045	731	552	1192	1192
(2) TD5	943	915	652	494	1035	1053
(3) TD6	871	846	612	461	964	975

Computation time for all 56 problems: (1) TD4, 19.4 min; (2) TD5, 18.7 min; (3) TD6, 18.7 min – in all cases using Dell Latitude D430, 1.2 GHz.

age travel speed increases though not at the same rate. Results for problem sets C1 and C2 are largely unchanged due to the binding constraint of the vehicle capacity. In the four types, the computation time for TD2 and TD3 is less than TD1 because when vehicles travel faster the number of time periods utilized (on average) to travel between customers is reduced.

Although in all cases, types (a)–(d), if the vehicles were to travel non-stop in the interval $[e_0, l_0]$ the vehicles would travel the same distance (extra 25%, 50%, and 75% more for speeds TD1, TD2, and TD3 respectively than in the original Solomon instances), the results in terms of fleet size are significantly different. Types (a) and (c) have significantly less vehicles than types (b) and (d).

Table 6
Distribution of time windows.

Fraction depot working time	0.0–0.2	0.2–0.4	0.4–0.6	0.6–0.8	0.8–1.0
Overlap counts between time windows and time periods					
R101	15	41	37	25	7
R201	26	48	33	29	18
C101	30	33	35	17	3
C201	24	27	27	27	16
RC101	17	60	53	31	5
RC201	26	50	39	29	16

To explain the differences in fleet size it necessary to understand how time windows are distributed in the Solomon problems. Table 6 presents the distribution of time window start, end times, and overlaps for the first instance in each problem type. Time window starts and ends are more frequent in the second and third period and less frequent at the extremes of the depot working time. Hence, speed distributions with the fasted periods at the beginning and end of the depot working time, type (b), or in the second half the working time, type (d), are less efficient that types (a) or (c) respectively.

7.3. Time dependent problems with soft time windows

The third set of results corresponds to the Solomon instances with time dependent travel speeds and soft time windows; results are presented in Table 7. In these instances the primary objective is to minimize number of vehicles, the secondary objective is to minimize time window violations, and the tertiary objective is to minimize the soft time window penalties and distance traveled. Table 7 presents the results in terms of fleet size, distance, and travel time.

The travel speed distributions TD4, TD5, and TD6 are listed in increasing order of travel speed variability. Without changing overall average speed, travel speed variability worsens the results in terms of number of vehicles for R1 and RC1 problems. Results in terms of distance traveled have little variation. Travel time slightly increases. Problem sets C1 and C2 are mostly unchanged because the binding constraint is vehicle capacity.

Although in all cases, types (a)–(d), if the vehicles were to travel non-stop in the interval $[e_0, l_0]$ the vehicles would travel the same distance (extra 25%, 50%, and 75% more for speeds TD1, TD2, and TD3 respectively than in the original Solomon instances), the results in terms of fleet size are significantly different. Types (a) and (c) have significantly less vehicles than types (b) and (d).

As customary in the VRP with time windows literature. Table 8 reports the number of soft time windows used, broken down into early and late service times as well as the penalty paid for early or late services. Usage of early soft time windows is more prevalent than the usage of late time windows. As expected, time window violations and penalties decrease as the number of vehicles used increases.

7.4. Time window elasticity

To better understand the response of the algorithm by problem type, a small perturbation is introduced. Given the importance of time windows, a small perturbation is introduced as follows:

$$e_i \leftarrow e_i + uniform[-1.0, 1.0] * 0.02 * (l_0 - e_0), \quad \forall i \in C$$

The perturbation is applied to problems TD1 type (a). The time window duration is not changed; hence, the difference $(l_i - e_i)$ is maintained for all customers as well as all the other customer characteristics such as demand level and location. It was

Table 7
VRPTW results – soft time windows.

Travel time distribution	R1	R2	C1	C2	RC1	RC2
Average number of vehicles by problem class						
(1) TD4	10.42	2.82	10.00	3.00	10.50	3.00
(2) TD5	10.42	2.64	10.00	3.00	10.63	3.00
(3) TD6	10.58	2.73	10.00	3.00	10.75	3.00
Average distance						
(1) TD4	1142	1010	856	666	1241	1135
(2) TD5	1131	1016	860	665	1226	1156
(3) TD6	1127	1016	869	660	1236	1149
Average travel time						
(1) TD4	1139	1023	871	669	1237	1150
(2) TD5	1134	1039	884	672	1220	1184
(3) TD6	1143	1061	938	685	1253	1213

Computation time for all 56 problems: (1) TD4, 19.5 min; (2) TD5, 19.6 min; (3) TD6, 19.4 min – in all cases using Dell Latitude D430, 1.2 GHz.

Table 8
VRPTW results – soft time windows.

Travel time distribution	R1	R2	C1	C2	RC1	RC2
Average number of soft time windows (early)						
(1) TD4	20.5	20.1	15.8	18.6	21.1	21.3
(2) TD5	20.4	19.9	18.6	14.9	22.1	21.1
(3) TD6	20.4	20.1	16.1	13.9	21.3	21.5
Average number of soft time windows (late)						
(1) TD4	18.0	13.6	8.2	17.0	16.3	14.1
(2) TD5	17.5	12.5	8.7	10.4	14.5	14.8
(3) TD6	15.8	12.5	6.2	8.4	15.1	15.1
Soft time window penalties (early)						
(1) TD4	386.6	1516.3	516.5	3025.5	381.7	1718.9
(2) TD5	425.1	1609.4	861.5	1467.7	448.5	1664.4
(3) TD6	419.3	1559.1	657.2	1697.8	446.2	1508.2
Soft time window penalties (late)						
(1) TD4	210.4	681.2	480.5	3267.1	197.8	695.9
(2) TD5	208.2	637.6	547.6	1797.7	173.2	692.1
(3) TD6	187.6	629.5	363.2	1708.8	189.2	787.8

mentioned in the previous section that the sum of perturbations are expected to be in absolute value equal to 1% of the depot working time and the expectation of the sum of all the perturbations is expected to be zero or "neutral" on average. To exemplify the magnitude of the change, for a 5-h depot working time a 1% change is equal to a 3 min change.

As an example, six different random seeds are employed to draw real numbers from the *uniform*[−1.0, 1.0] distribution; clearly, more seeds can be used but it is essential that the researcher publishes the results of *all* the seeds to avoid bias. The results, presented in Table 9, show that a small perturbation can have a significant impact even on the *average* number of

Table 9
Solution quality elasticity – TD1 type (a).

	R1	R2	RC1	RC2
Average number of vehicles				
Random seed 1	11.67	2.73	11.38	3.25
Random seed 2	11.58	2.82	11.38	3.25
Random seed 3	11.50	2.82	11.38	3.25
Random seed 4	11.92	2.91	11.25	3.25
Random seed 5	11.50	2.73	11.25	3.25
Random seed 6	11.58	2.82	11.38	3.25
Average distance				
Random seed 1	1261	1218	1410	1383
Random seed 2	1299	1225	1413	1427
Random seed 3	1285	1202	1424	1411
Random seed 4	1271	1196	1395	1425
Random seed 5	1276	1240	1411	1412
Random seed 6	1264	1227	1406	1435
Average travel time				
Random seed 1	1045	992	1161	1112
Random seed 2	1079	998	1170	1148
Random seed 3	1069	977	1171	1142
Random seed 4	1059	977	1160	1154
Random seed 5	1061	1003	1170	1148
Random seed 6	1052	999	1164	1167

VRPTW results – hard time windows – type (a)

Average number of vehicles by problem class

Travel time distribution	R1	R2	C1	C2	RC1	RC2
TD1 – unperturbed	11.67	2.82	10.00	3.00	11.38	3.25

Average distance

Travel time distribution	R1	R2	C1	C2	RC1	RC2
TD1– unperturbed	1295	1216	879	657	1405	1444

Average travel time

Travel time distribution	R1	R2	C1	C2	RC1	RC2
TD1– unperturbed	1080	990	729	563	1164	1177

Table 10
Key statistics number of vehicles.

Statistic	R1	R2	RC1	RC2
Min	11.50	2.73	11.25	3.25
Max	11.92	2.91	11.38	3.25
Average	11.63	2.80	11.33	3.25
Mean	11.58	2.82	11.38	3.25
St. dev.	0.16	0.07	0.06	0.00
Coef. of variation	1.3%	2.4%	0.6%	0.0%

vehicles for problem type R1. The impact or magnitude of the changes is less for problems RC1 and R2 and problems RC2 are not affected *on average*. The comparable unperturbed results from Table 2 are reproduced at the bottom to facilitate the comparison. Problems C1 and C2 are not included because in these problems fleet size is not sensitive to changes in travel speed or time windows.

The results obtained in Table 9 are obtained with exactly the same algorithm, parameters, and running time employed to obtain the results presented in Table 2. For problem types R1, R2, and RC1 small perturbations can lead to outcomes that are, on average, roughly ±1% to 3% in terms of fleet size. Unfortunately, these values cannot be compared to other solution algorithms because, to the best of the author's knowledge, this type of robustness or elasticity analysis has never been performed for VRP problems with time windows or time dependent travel times. Clearly, performance on individual instances is important to measure the overall quality of the algorithm. But, as expressed by Barr et al. (1995) and Golden et al. (1998) the gain in solution quality should not come at the expense of over fitting the solution approach to the known instance since in real-world applications instances do change regularly or extremely often in real-time applications. A balanced approach to testing VRP algorithms should include absolute solution quality as well as elasticity to changes in small data inputs.

Table 10 shows some descriptive statistics including minimum, maximum, median, average, standard deviation, and coefficient of variation (CV). Although problems R1 show the highest absolute change, percentage wise the problems R2 show the greater change in terms of CV. Only time window results are shown since small perturbations to customer demand did not produce any change in average fleet size (zero elasticity). The elasticity to perturbations is also important when comparing algorithms and it will facilitate a better understanding of VRP algorithms. For example, a small difference in average RC2 performance can be statistically significant whereas the same absolute different for R1 problems may not be statistically significant.

Some researchers have already suggested the use of statistical tests to compare algorithms. For example, Taillard (Taillard, 2001; Golden et al., 1998) suggested the use of the non-parametric Mann–Whitney test to estimate whether the distribution of the results produced by each algorithm are of comparable quality. For paired results, e.g. same instances, the Wilcoxon test is recommended. To have a fair comparison, the results to be compared must have similar running times and in similar computers. Ideally, the algorithms would be tested on a meaningful range of running times (short and long).

8. Computational complexity

The relative simplicity of the IRCI allows for a straightforward algorithmic analysis. The auxiliary heuristic $\mathbf{H}_r$ is called by the construction algorithm no more than $nW|\mathbf{\Delta}|$ times; where n is the number of customers. Hence, the worst case number of operations of the construction algorithm is of order $(nW|\mathbf{\Delta}|O(\mathbf{H}_r(n)))$ where $O(\mathbf{H}_r(n))$ denotes the computational complexity of the auxiliary algorithm to route n customers. Hence, the complexity and running time of the auxiliary heuristic $\mathbf{H}_r$ will have a substantial impact on the overall running time.

The improvement procedure calls the construction procedure a finite number of times. The number of calls is bounded by the number of routes $|R| = m$. Let n_i be largest number of customers contained a subset of routes u that is improved in each iteration of $\mathbf{H}_i$. The computational complexity of a call to the construction algorithm is then $(n_iW|\mathbf{\Delta}|O(\mathbf{H}_r(n_i)))$. The complexity of the $\mathbf{H}_i$ algorithm is then of order $O(mn_iW|\mathbf{\Delta}|O(\mathbf{H}_r(n_i)))$ where $n_i < n$ if $u < m$.

If constant speed intervals are used to represent time dependent speeds and the depot working time $[e_0, l_0]$ is partitioned into p time periods, the computational complexity of the service start time algorithms, $\mathbf{H}_{yb}$ and $\mathbf{H}_{yf}$ is of order $O(np)$. Each travel time calculation between any two customers has a computational complexity $O(p)$ – see Appendix A.

Table 11
VRPTW average run time ratios – TD3.

n	Ratio	$O(n^2)$	$O(n^3)$	% $O(n^3)$
25	1.0	1	1	100
50	3.3	4	8	41
100	17.4	16	64	27
200	90.5	64	512	18

To test the increase in computational running time, instances with different numbers of customers are run. Firstly, the first 25 and 50 customers of each Solomon problem are taken to create instances with $n = 25$ and $O(np) = 50$ respectively. Secondly, to create and instance with $n = 200$ customers, for each customer in the original Solomon problem a "clone" is created. Each "clone" has the same customer characteristics of an "original" customer but multiplied by a random number drawn from [0.95, 1.05] to avoid exact replicas. Hence, the clones are "similar" but not "identical".

The summary results for each problem size are shown in Table 11. The results are expressed as the ratio between each average running time and the running time for $n = 25$. To facilitate comparisons, the corresponding increases in running time ratios for $O(n^2)$ and $O(n^3)$ are also presented.

The results indicate that the average running time is increasing by a factor of $O(n^2)$. This is expected from the complexity analysis as the complexity of the nearest neighbor heuristic $\mathbf{H}_r$ has a worst case of $O(n^2)$. As customer size n increases, the ratio as a % of the n^3 growth factor is decreasing – see last column of Table 11.

8.1. Travel time updates

Current location and communication systems allow the utilization of real-time information about vehicle locations and demands to update routes whenever there are significant changes in travel times (Regan et al., 1996). For example, delays due to non-recurrent events such as accidents can significantly alter travel times between customers. In these situations fast algorithms may have a significant edge over significantly slower though higher quality approaches. The advantage of fast algorithms is more significant when there are time windows or significant differences among customers' time and price sensitivity (Kim et al., 2004).

The route improvement algorithm proposed in this research can be easily extended to situations where a subset of arc travel times is updated as follows:

Algorithm $\mathbf{H}_u$

1. Find the subset of $\mathbf{T} = T_1, T_2, \ldots, T_p$ time periods that are affected by the update.
2. Find the set of $t_{ij}(b_i)$ values that has been updated.
3. Select the subset of customers $C(updated)$ that are affected by changes in travel times.
4. Form a subset of routes $R(updated)$ that contains all customers in $C(updated)$.
5. Apply the route improvement algorithm $\mathbf{H}_i$ to each route contained in $R(updated)$.

The running time of the update algorithm $\mathbf{H}_u$ will be a function of the number of customers and routes affected by the new travel times: $O(mn_iW|\Delta|O(\mathbf{H}_r(n_i)))$ where $n_i < n$ if $u < m$. An objective function with soft time windows guarantees that a solution will be found after the first run of $\mathbf{H}_u$. The evaluation of real-time strategies is beyond the scope of this research. The interested reader is referred to research in real-time environments, e.g. Haghani and Jung (2005).

9. Conclusions

This is the first research effort to publish solutions to time dependent problems with hard time windows using standard and replicable instances that simulate variations in travel times that can be found in congested urban areas. The results show that the overlap between the temporal distribution of the congested periods and the distribution of customer time windows has a significant impact on routing costs. Results also indicate that a constant travel speed assumption in congested urban areas with customer hard time windows produce suboptimal or unfeasible solutions. In addition, a method to study the impact of perturbations by problem type is introduced and results showed that problem types have significantly different levels of time window elasticity to small perturbations in the data inputs.

The proposed solution algorithm can tackle constant speed or time-dependent speed problems with hard or soft time windows without any alteration in their structure. This is the first research effort to published results that can be readily benchmarked for TDVRP with hard time windows and it provides faster and higher quality results than already published methods to solve the TDVRP with hard time windows in constant speed Solomon instances. The analysis and experimental results of the computational complexity indicate that average computational time increases proportionally to the square of the number of customers. The relative low computational complexity, simplicity, and generality of the IRCI are important factors in real-world applications with constant and time dependent travel times. Future research efforts may explore alternative grouping methodologies, alternative route construction approaches, and real-time implementations.

Acknowledgements

The author gratefully acknowledges the Oregon Transportation, Research and Education Consortium (OTREC) and the Department of Civil & Environmental Engineering in the Maseeh College of Engineering & Computer Science at Portland State University for sponsoring this research. The author would like to thank Stuart Bain, at the University of Sydney, for his assistance coding during the initial stages of this research and Myeonwoo Lim, Computer Science Department at Portland State

University, for assistance coding during the final stages of this research. The author gratefully acknowledges Brian Davis and anonymous reviewers for their useful suggestions and comments. Any errors are the sole responsibility of the author.

Appendix A

Unlike the algorithms presented in Section 4, the calculation of travel times is dependent on the specific data format and speed functions. Travel times from any two given customers i and j are calculated using an iterative forward calculation from the arrival time at customer i. The depot working time $[e_0, l_0]$ is partitioned into p time periods $\mathbf{T} = T_1, T_2, \ldots, T_p$; each period T_k has an associated constant travel speed s_k. The algorithm is adapted from Ichoua et al. (2003):

Data:
  $\mathbf{T} = T_1, T_2, ..., T_p$, and corresponding travel speeds
  v_i, v_j, a_i: Given any two customers and the arrival time to customer i
START
if $a_i < e_i$ **then**
  $b_i \leftarrow e_i + g_i$
else $b_i \leftarrow a_i + g_i$
end if
find k, $t_{\underline{k}} \leqslant b_i \leqslant t_{\bar{k}}$
$a_j \leftarrow b_i + d_{ij}/s_k$
$d \leftarrow d_{ij}$, $t \leftarrow b_i$
while $a_j > t_{\bar{k}}$ **do**
  $d \leftarrow d - (t_{\bar{k}} - t)s_k$
  $t \leftarrow t_{\bar{k}}$
  $a_j \leftarrow t + d/s_{k+1}$
  $k \leftarrow k + 1$
end while
Output:
  a_j, arrival time at customer j
END $\mathbf{H}_r$

The algorithm is guaranteed to find the arrival time in no more than p iterations.

References

Ahn, B.H., Shin, J.Y., 1991. Vehicle-routing with time windows and time-varying congestion. Journal of the Operational Research Society 42 (5), 393–400.
Balakrishnan, N., 1993. Simple heuristics for the vehicle routeing problem with soft time windows. The Journal of the Operational Research Society 44 (3), 279–287.
Barr, R.S., Golden, B.L., Kelly, J.P., Resende, M.G.C., Stewart, W.R., 1995. Designing and reporting on computational experiments with heuristic methods. Journal of Heuristics 1 (1), 9–32.
Braysy, I., Gendreau, M., 2005a. Vehicle routing problem with time windows, part 1: route construction and local search algorithms. Transportation Science 39 (1), 104–118.
Braysy, I., Gendreau, M., 2005b. Vehicle routing problem with time windows, part II: metaheuristics. Transportation Science 39 (1), 119–139.
Chiang, W.C., Russell, R.A., 2004. A metaheuristic for the vehicle-routeing problem with soft time windows. Journal of the Operational Research Society 55 (12), 1298–1310.
Cordeau, J.F., Gendreau, M., Laporte, G., Potvin, J.Y., Semet, F., 2002. A guide to vehicle routing heuristics. Journal of the Operational Research Society 53 (5), 512–522.
Dabia, S., Van Woensel, T., De Kok, A.G., 2010. A Dynamic Programming Approach to Multi-Objective Time-Dependent Capacitated Single Vehicle Routing Problems with Time Windows. Working Paper 313, Eindhoven University of Technology, School of Industrial Engineering, The Netherlands.
Desrochers, M., Lenstra, J.K., Savelsbergh, M.W.P., Soumis, F., 1988. Vehicle routing with time windows: optimization and approximation. Vehicle Routing: Methods and Studies 16, 65–84.
Desrosiers, J., Dumas, Y., Solomon, M.M., Soumis, F., 1995. Time constrained routing and scheduling. In: Ball, M., Magnant, T., Monma, C., Nemhauser, G. (Eds.), Handbooks in Operations Research and Management Science Network Routing, vol. 8. North-Holland.
Donati, A.V., Montemanni, R., Casagrande, N., Rizzoli, A.E., Gambardella, L.M., 2008. Time dependent vehicle routing problem with a multi ant colony system. European Journal of Operational Research 185 (3), 1174–1191.
Dongarra, J.J., 2007. Performance of Various Computers Using Standard Linear Equations Software. Technical Report CS-89-85, University of Tennessee, November 20, 2007. <http://www.netlib.org/utk/people/JackDongarra/papers.htm> (accessed 23.11.07).
Figliozzi, M., 2008. An iterative route construction and improvement algorithm for the vehicle routing problem with soft and hard time windows. In: Applications of Advanced Technologies in Transportation (AATT) 2008 Conference Proceedings, Athens, Greece, May, 2008.
Figliozzi, M.A., 2009. A route improvement algorithm for the vehicle routing problem with time dependent travel times. In: Proceeding of the 88th Transportation Research Board Annual Meeting, Washington, DC, January, 2009. <http://web.cecs.pdx.edu/~maf/publications.html>.
Figliozzi, M.A., 2010a. The impact of congestion on commercial vehicle tours and costs. Transportation Research Part E: Logistics and Transportation 46E (4), 496–506.
Figliozzi, M.A., 2010b. An iterative route construction and improvement algorithm for the vehicle routing problem with soft-time windows. Transportation Research Part C: Emerging Technologies 18 (5), 668–679.
Fleischmann, B., Gietz, M., Gnutzmann, S., 2004. Time-varying travel times in vehicle routing. Transportation Science 38 (2), 160–173.
Golden, B., Wasil, E., Kelly, J., Chao, I., 1998. Metaheuristics in vehicle routing. In: Craignic, T., Laporte, G. (Eds.), Fleet Management and Logistics. Kluwer, Boston.

Haghani, A., Jung, S., 2005. A dynamic vehicle routing problem with time-dependent travel times. Computers and Operations Research 32 (11), 2959–2986.
Hill, A.V., Benton, W.C., 1992. Modeling intra-city time-dependent travel speeds for vehicle scheduling problems. Journal of the Operational Research Society 43 (4), 343–351.
Ichoua, S., Gendreau, M., Potvin, J.Y., 2003. Vehicle dispatching with time-dependent travel times. European Journal of Operational Research 144 (2), 379–396.
Jung, S., Haghani, A., 2001. Genetic algorithm for the time-dependent vehicle routing problem. Transportation Research Record 1771 (-1), 164–171.
Kim, Y.J., Mahmassani, H.S., Jaillet, P., 2004. Dynamic truckload routing, scheduling, and load acceptance for large fleet operation with priority demands. Transportation Research Record 1882, 120–128.
Kok, A.L., 2010. Congestion Avoidance and Break Scheduling within Vehicle Routing. Ph.D. Thesis, University of Twente, Enschede, The Netherlands.
Malandraki, C., 1989. Time Dependent Vehicle Routing Problems: Formulations, Solution Algorithms and Computational Experiments. Ph.D. Dissertation, Northwestern University, Evanston, Illinois.
Malandraki, C., Daskin, M.S., 1992. Time-dependent vehicle-routing problems – formulations, properties and heuristic algorithms. Transportation Science 26 (3), 185–200.
Malandraki, C., Dial, R.B., 1996. A restricted dynamic programming heuristic algorithm for the time dependent traveling salesman problem. European Journal of Operational Research 90 (1), 45–55.
Regan, A., Mahmassani, H., Jaillet, P., 1996. Dynamic decision making for commercial fleet operations using real-time information. Transportation Research Record 1537, 91–97.
Schrimpf, G., Schneider, J., Stamm-Wilbrandt, H., Dueck, G., 2000. Record breaking optimization results using the ruin and recreate principle. Journal of Computational Physics 159 (2), 139–171.
SINTEF, 2008. Benchmarks – Vehicle Routing and Travelling Salesperson Problems. SINTEF Applied Mathematics, Department of Optimization, Norway, <http://www.top.sintef.no/vrp/benchmarks.html>.
Soler, D., Albiach, J., Martínez, E., 2009. A way to optimally solve a time-dependent vehicle routing problem with time windows. Operations Research Letters 37 (1), 37–42.
Solomon, M.M., 1987. Algorithms for the vehicle-routing and scheduling problems with time window constraints. Operations Research 35 (2), 254–265.
Taillard, E., 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), 170–186.
Taillard, E.D., 2001. Comparison of Non-Deterministic Iterative Methods, Citeseer.
Van Woensel, T., Kerbache, L., Peremans, H., Vandaele, N., 2008. Vehicle routing with dynamic travel times: a queueing approach. European Journal of Operational Research 186 (3), 990–1007.