1-2011

Freight Distribution Problems in Congested Urban Areas: Fast and Effective Solution Procedures to Time-Dependent Vehicle Routing Problems

Miguel A. Figliozzi
Portland State University, figliozzi@pdx.edu

Citation Details

Figliozzi, Miguel. Freight Distribution Problems in Congested Urban Areas: Fast and Effective Solution Procedures to Time-Dependent Vehicle Routing Problems. OTREC-RR-11-05. Portland, OR: Transportation Research and Education Center (TREC), 2011. http://dx.doi.org/10.15760/trec.108

OTREC

OREGON TRANSPORTATION RESEARCH AND EDUCATION CONSORTIUM

FINAL REPORT

Freight Distribution Problems in Congested Urban Areas: Fast and effective solution procedures to time-dependent vehicle routing problems

OTREC-RR-11-05
January 2011

FREIGHT DISTRIBUTION PROBLEMS IN CONGESTED URBAN AREAS: FAST AND EFFECTIVE SOLUTION PROCEDURES TO TIME-DEPENDENT VEHICLE ROUTING PROBLEMS

Final Report

OTREC-RR-11-05

by

Miguel Figliozzi, Associate Professor
Civil and Environmental Engineering
Portland State University

for

OTREC
OREGON TRANSPORTATION RESEARCH AND EDUCATION CONSORTIUM

P.O. Box 751
Portland, OR 97207

January 2011

Technical Report Documentation Page

1. Report No. **OTREC-RR-11-05**	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle Freight Distribution Problems in Congested Urban Areas: Fast and effective solution procedures to time-dependent vehicle routing problems		5. Report Date January 2011
		6. Performing Organization Code
7. Author(s) Miguel Figliozzi, Portland State University, Civil and Environmental Engineering, P.O. Box 751, Portland, OR, 97207		8. Performing Organization Report No.
9. Performing Organization Name and Address		10. Work Unit No. (TRAIS)
		11. Contract or Grant No.
12. Sponsoring Agency Name and Address Oregon Transportation Research and Education Consortium (OTREC), P.O. Box 751, Portland, Oregon 97207		13. Type of Report and Period Covered
		14. Sponsoring Agency Code
15. Supplementary Notes		

16. Abstract

Congestion is a common phenomenon in all medium to large cities of the world. Reliability of freight movement in urban areas is an important issue to manufacturing or service companies whose operation is based in just-in-time approaches. These companies tend to provide high value or time sensitive products/services. As congestion increases, carriers face increasing challenges to satisfy their time sensitive customers in an economical way. Route designs or schedules which require long computation times or ignore travel time variations will result in inefficient and suboptimal solutions. Poorly designed routes that lead freight vehicles into congested arteries and streets not only increases supply chain and logistics costs but also exacerbate externalities associated with freight traffic in urban areas such as greenhouse gases, air pollution, noise, and accidents.

Whilst it is rarely possible to entirely avoid the impacts of congestion, it is feasible to schedule operations so that the effects of congestion are minimized. Better scheduling can be effectively supported by the advent of inexpensive and ubiquitous Information and Communication Technologies (ICT). The use of mobile phone technology and on-board routing devices allows fluid communication between truck drivers and fleet operators in real-time. In such a real time operation it becomes possible to dynamically reassign vehicles, including modifying the order in which customers are served and diverting a vehicle already en-route to service another customer. However, without fast routing methods that can take advantage of real time congestion information carriers cannot reap the benefits of real-time information.

From the operational point of view, congestion creates a substantial variation in travel times during peak morning and evening hours. This is problematic for all vehicle routing models which rely on a constant value to represent vehicle speeds. And while the ubiquitous availability of real time traffic information allows drivers to reactively alter routes and customer service sequences to better cope with congestion, static routing models are unable to take advantage of these advances in real-time information provision in order to proactively find adequate routing solutions. In addition, changes in travel time caused by congestion cannot be accurately represented in static models.

Research in time-dependent vehicle routing problem is comparatively meager and current solution methods are inadequate for practical carrier operations which need to provide fast solutions for medium to large instances. Even faster solution methods are essential to take advantage of real time information. The major aim of this proposal is to develop and evaluate new methods for vehicle routing in congested urban areas. The emphasis will be placed on improving the running time of the existing methods.

17. Key Words Vehicle Routing Problem, Time-dependent, Congestion, Algorithms		18. Distribution Statement No restrictions. Copies available from OTREC: www.otrec.us	
19. Security Classification (of this report) Unclassified	20. Security Classification (of this page) Unclassified	21. No. of Pages 53	22. Price

ACKNOWLEDGEMENTS

The author gratefully acknowledges the Oregon Transportation Research and Education Consortium (OTREC) for sponsoring this project. This work was also supported by the Department of Civil and Environmental Engineering in the Maseeh College of Engineering and Computer Science at Portland State University. The author also is thankful to the reviewers and editors for their helpful comments and suggestions, and to research assistants Ryan Conrad and Myeonwoo Lim for their comments. Any errors or omissions are the sole responsibility of the author.

TABLE OF CONTENTS

LIST OF TABLES

EXECUTIVE SUMMARY

Congestion is a common phenomenon in medium and large cities worldwide. Reliability of freight movement in urban areas is an important issue for manufacturing or service companies whose operation is based on just-in-time approaches. These companies tend to provide high-value or time-sensitive products/services. As congestion increases, carriers face increasing challenges to satisfy their time-sensitive customers in an economical way.

In urban areas, congestion creates a substantial variation in travel speeds during peak morning and evening hours. Route designs or schedules which require long computation times or ignore travel-time variations will 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 exacerbate externalities associated with freight traffic in urban areas, such as greenhouse gases, air pollution, noise and accidents. Better scheduling can be effectively supported by the advent of inexpensive and ubiquitous Information and Communication Technologies (ICT). However, without fast-routing methods that can take advantage of real-time congestion information, carriers cannot reap the benefits of real-time information.

This research presents a new approach, an iterative route construction and improvement algorithm (IRCI), for the time-dependent vehicle routing problem (TDVRP) with hard or soft time windows. Improvements are obtained at a *route* level; hence the proposed approach does not rely on any type of local improvement procedure.

Firstly, the IRCI algorithm is proposed to sequentially solve vehicle routing problems with soft time windows (VRPSTW) and hard time windows (VRPHTW). Secondly, the solution algorithm is adapted to tackle time-dependent speed problems without any alteration in their structure. A new formulation for the TDVRP with soft and hard time windows is presented. Leveraging on the well-known Solomon instances, new test problems that capture the typical speed variations of congested urban settings are proposed. Results in terms of solution quality as well as computational time are presented and discussed.

Due to its modular and hierarchical design, the IRCI algorithm is intuitive, easy to code, and able to accommodate general cost and penalty functions. The solution quality and computational time of the new algorithm is compared against existing results on benchmark problems for the VRPHTW and VRPSTW. Furthermore, the algorithm can be used to obtain faster simultaneous solutions for both VRPHTW and VRPSTW problems using the soft time-windows solution as a lower bound for hard time-window problems. Despite its simplicity and flexibility, the algorithm performs well in terms of solution quality and speed in instances with soft and hard time windows.

The computational complexity of the IRCI is analyzed and experimental results indicate that average computational time increases proportionally to the square of the number of customers. Furthermore, the proposed algorithm is remarkably faster than local search heuristics in terms of running time and computational complexity.

1.0 INTRODUCTION

Congestion is a common phenomenon in most urban areas worldwide. 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 exacerbate externalities associated with freight traffic in urban areas, such as greenhouse gases, air pollution, noise and accidents. Travel time between customers and depots is found to be a crucial factor that exacerbates the negative impacts of congestion; congestion also affects carriers' cost structure and the relative weight of wages and overtime expenses (*Figliozzi, 2009*).

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.

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 time-dependent problems, (c) introduce readily replicable time-dependent instances and analyze the computational results, and (d) analyze the computational complexity of the solution approach.

This report is organized as follows: Section 2.0**Error! Reference source not found.** provides a literature review for the problems with constant speed and soft/hard time windows; Section 3.0 introduces the mathematical notation and describes the new iterative route construction and improvement (IRCI) algorithm; Section 4.0 compares IRCI computation time and solution quality against existing solutions available in the literature for instances with constant speed and soft/hard time windows; Section 5.0 discusses IRCI algorithmic properties; Section 6.0 introduces a literature review for the TDVRP; Section 7.0 introduces notation for the TDVRP and formulates the problem for congested environments; Section 8.0 presents the additional elements of IRCI to solve time-dependent routing problems and minimize fleet size; Section 9.0 presents benchmark problems for the TDVRP; and Section 10.0 discusses computational results. Section 11.0 analyses the worst-case and average computational complexity of the TDVRP algorithm, and Section 12.0 concludes the report.

2.0 LITERATURE REVIEW FOR THE VRPHTW AND VRPSTW

Heuristics to solve the VRPHTW can be classified – in increasing order of solution quality – as construction heuristics, local search heuristics, and metaheuristics. Although metaheuristics generally produce solutions of higher quality, this is usually at the expense of significantly longer computation times. There is a clear tradeoff between computation time and solution quality.

Route construction algorithms work by inserting customers one at a time into partial routes until a feasible solution is obtained. Construction heuristics include the work of Solomon (1987), Potvin and Rousseau (1993), and Ioannou et al. (2001). Local search methods improve on feasible solutions performing exchanges within a neighborhood while maintaining the feasibility of the solutions. Some of the most successful local improvement methods include the algorithms proposed by Russel (1995), Caseau and Laburthe (1999), Cordone and Calvo (2001), and Braysy (2002).

Metaheuristics include a diverse set of methods such as simulated annealing, genetic algorithms, tabu search, ant-colony, and constraint programming. Some of the most successful methaheuristics include the algorithms proposed by Taillard et al. (1997), Liu and Shen (1999), Homberger and Gehring (1999), Berger et al. (2003), and Braysy (2003). For additional references and a review of the large body of VRPHTW research, the reader is referred to a recent comprehensive survey by Braysy and Gendreau (2005a,b).

The body of work related to the VRPSTW is relatively scant. Early work on the topic includes the work of Sexton and Choi (1986) using Benders decomposition to solve a single-vehicle pickup and delivery routing problem. Ferland and Fortin (1989) solves variations of the VRPSTW where customers' time windows are adjusted to lower service costs. Koskosidis et al. (1992) propose a generalized assignment problem of customers to vehicles and a series of traveling salesman problems with soft time-window constraints.

Balakrishnan (1993) proposes construction heuristics for the VRPSTW based on the nearest neighbor, Clarke and Wright savings, and space–time rules algorithms. The heuristics are tested on a subset of the Solomon set problems for hard time windows using linear penalty functions. Taillard et al. (1997) propose a tabu search heuristic to solve a VRPSTW as proposed by Balakrishnan (i.e., with linear penalty functions). The tabu search algorithm produced very good results on the Solomon set with hard time windows; however, no results are reported for the VRPSTW.

Ioannou et al. (2003) solves Solomon problems and extended Solomon problems of up to 400 customers with a nearest neighbor that generate and modify customer time windows to find lower cost solutions; no computation times are reported. Chiang and Russell (2004) uses a tabu search approach with a mixed neighborhood structure and advance recovery to find some of the best solutions ever reported for Solomon VRPSTW instances. The algorithm designed by Ibaraki et al. (2005) is another metaheuristic that could handle soft time-window constraints and penalties using a local search based on a cyclic-exchange neighborhood to assign and sequence customers; only results for instances with hard time windows are reported . Calvete et al. (2007) propose a goal programming approach to the vehicle routing and solve medium-size problems (less than 70 customers) with soft and hard time windows, a heterogeneous fleet of vehicles, and multiple objectives.

As indicated by Braysy and Gendreau (2005a,b), fair and meaningful comparisons of vehicle routing heuristics require standard benchmark problems and the full reporting of (a) solution quality, (b) number of runs needed and computation time per run, and (c) computing power or processor speed. From the survey of the VRPSTW, only two journal publications comply with these prerequisites: Balakrishnan (1993) and Chiang and Russell (2004). Regarding VRPHTW, only Taillard et al. (1997) and Ibaraki et al. (2005) present algorithms that are designed to handle soft and hard time windows and also comply with the reporting of solution quality, computation time and processor speed.

3.0 SOLUTION ALGORITHM FOR VRPHTW AND VRPSTW

This section first introduces a precise mathematical definition of the VRPHTW and VRPSTW studied in this research. The remainder of this section is to describe the solution algorithm.

3.1 PROBLEM DEFINITION

The vehicle routing problem with hard time windows (VRPHTW) studied in this research can be described as follows. Let $G = (V, A)$ be a graph where $V = (v_0, ..., v_n)$ is a vertex set and $A = \{(v_i, v_j) : i \neq j \wedge i, j \in V\}$ is an arc set. Vertex v_0 denotes a depot at which the routes of m identical vehicles of capacity $q_{\max}$ start and end. The set of vertices $C = \{v_1, ..., v_n)$ specify the location of a set of n customers. Each vertex in V has an associated demand $q_i \geq 0$, a service time $s_i \geq 0$, and a service time window $[e_i, l_i]$. Each arc (v_i, v_j) has an associated constant distance $d_{ij} > 0$ and travel time $t_{ij} > 0$. The arrival time of a vehicle at customer $i, i \in C$ is denoted a_i and its departure time b_i; the beginning of the service time is denoted y_i. The primary objective function for the VRPHTW is the minimization of the number of routes. A secondary objective is the minimization of total time or distance. The solution to the VRPHTW must satisfy the following:

(a) the value of m is not specified initially, it is an output of the solution algorithm;

(b) a route cannot start before e_0 and cannot end after l_0;

(c) service to customer i cannot start before e_i and cannot start after l_i;

(d) every route starts and ends at the depot v_0;

(e) every customer is visited exactly once by one vehicle; and,

(f) the total demand of any vehicle route does not exceed the vehicle capacity.

The VRPSTW is a relaxation of the VRPHTW. With soft time windows, there is an allowable violation of time windows denoted $P_{\max} \geq 0$. The time window of each customer $i, i \in C$ can be enlarged to $[e_i - P_{\max}, l_i + P_{\max}] = [e_i^{\#}, l_i^{\#}]$. In addition, an early penalty $p_e(e_i - y_i)$ is applied if service time starts early (i.e., $y_i \in [e_i^{\#}, e_i]$). Similarly, a late penalty $p_l(y_i - l_i)$ is applied if service starts late (i.e., $y_i \in [l_i, l_i^{\#}]$). The primary objective function for the VRPSTW is the minimization of the number of routes. A secondary objective is the minimization of the number of time-window violations. A third objective is the minimization of total time or distance plus penalties for early or late deliveries. 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.

It is commonly assumed in the literature that fixed costs associated with each additional route (vehicle) outweigh travel time or distance-related costs. As discussed in Section 5, the presented IRCI algorithm can be applied to any hard or soft time-window problem with an objective function that is a combination of positive functions of fleet size, travel time, travel distance, and early/late penalties.

3.2 SOLUTION ALGORITHMS

The solution method is divided into two phases: route construction and route improvement. The route-construction phase utilizes two algorithms: (a) an auxiliary route-building algorithm and (b) a route-construction algorithm. The route-improvement phase also utilizes two algorithms: (c) a route-improvement algorithm and (d) a service-time improvement algorithm. Using a bottom-up approach, the algorithms are introduced in the following order: (a) the auxiliary algorithm, (b) the construction algorithm, and (c) the route-improvement algorithm.

3.2.1 The Auxiliary Algorithm

The auxiliary routing algorithm $\mathbf{H}_{\mathrm{r}}$ 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 VRPHTW or VRPSTW.

In this research $\mathbf{H}_{\mathrm{r}}$ is a generalized nearest neighbor heuristics (GNNH). The GNNH has four inputs: (a) the weights or parameters for "generalized cost" function denoted by $\Delta = \{\delta_0, \delta_1, ..., \delta_i\}$, (b) an initial vertex denoted by v_i, (c) a set of customers to route denoted by C, and (d) a depot location denoted by v_0. The GNNH starts every route by finding the unrouted customer with the least appending "generalized cost." At every subsequent iteration, the heuristics searches for the remaining unrouted customer with the least appending cost.

The "generalized cost" function used in this research accounts for geographical and temporal closeness among customers, the remaining capacity in the vehicle, and the cost of adding a new vehicle if the next customer is infeasible. Let i denote the initial vertex and let j denote the customer to append next. Let q_i denote the remaining capacity of the vehicle after serving customer i. The service at a customer $i, i \in V$ begins at time $y_i = \max(a_i, e_i)$. The generalized cost of going from customer i to customer j is estimated as:

$$\mathrm{g}(\Delta, i, j) = \delta_1 d_{ij} + \delta_2 (a_j - (a_i + s_i)) + \delta_3 (l_j - (a_i + s_i + t_{ij})) + \delta_4 (q_i - d_j)$$

The parameter δ_2 takes into account the "slack" between the completion of service at i and earliest feasible beginning of service at j, i.e. $a_j = \max(y_i + s_i + t_{ij}, e_j)$. Following Solomon's approach (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 start. The parameter δ_4 is introduced in this research and takes into account the capacity slack (*vehicle capacity that is still unused*) of the vehicle after serving customer j.

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:

$$\mathrm{g}(\Delta, i, j) = \delta_0 + \delta_1 d_{0j} + \delta_2 a_j + \delta_3 (l_j - t_{0j}) + \delta_4 (q_{\max} - d_j)$$

The parameter δ_0 is the cost of adding a new vehicle. The same GNNH can be applied to VRPSTW with the addition of two terms. For feasible customers:

$$\begin{aligned}\mathrm{g}(\Delta, i, j) = \delta_1 d_{ij} + \delta_2 (a_j - (a_i + s_i)) + \delta_3 (l_j - (a_i + s_i + t_{ij})) + \delta_4 (q_i - d_j) + \\ + \delta_5 [e_j - a_j]^+ + \delta_6 [a_j - l_j]^+\end{aligned}$$

The parameters δ_5 and δ_6 are added to account for possible early or late service penalties, respectively; for infeasible customers δ_0 is added. With soft time windows, the service at a customer $i, i \in V$ begins at time $y_i = \max(a_i, e_i^\#)$. For problems with general time windows (i.e., two or more time window intervals), the generalized cost is calculated for each time interval and the least expensive interval provides the generalized cost for that particular customer.

The auxiliary route heuristic is defined as $\mathbf{H}_\mathrm{r}(\Delta, v_i, C, v_0)$ where $\Delta = \{\delta_0, \delta_1, ..., \delta_6\}$ 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 that starts at v_i. In all cases, the deltas are positive weights that satisfy: $\delta_1 + \delta_2 + \delta_3 = 1$ and $\delta_i \geq 0 \; i \in \{0, 1, ..., 6\}$.

3.2.2 The Route Construction Algorithm

In this algorithm, denoted $\mathbf{H}_\mathrm{c}$, routes are constructed sequentially. Given a partial solution and a set of unrouted customers, the algorithm uses the auxiliary heuristic $\mathbf{H}_\mathrm{r}$ to search for the feasible least-cost set of routes. The algorithm also uses an auxiliary function $\mathrm{w}(v_i, C, \mathrm{g}, W)$ that given a set of unrouted customers C, a vertex $v_i \notin C$, and a generalized cost function $\mathrm{g}(\Delta, v_i, v_j)$ returns a set of vertexes with the lowest generalized costs $\mathrm{g}(\Delta, v_i, v_j)$ for all $v_j \in C$.

Functions or Algorithms:

$\mathbf{H}_\mathrm{r}$: Route building heuristic.

$\mathrm{w}(v_i, C, \mathrm{g}, W)$: returns set of vertexes with the lowest generalized costs

Data:

C : Set of customers to route (not including the depot v_0)

LLimit = initial number of routes or best-known lower bound

W : Width of the search determined by the user, number of solutions to be built and compared before adding a customer to a route

$\mathbf{\Delta}$: space of the route heuristic generalized cost-function parameters

```
START $\mathbf{H}_{\mathrm{c}}$
 $start \leftarrow v_0$
 $start \leftarrow v_0$
 $bestSequence \leftarrow v_0$
 $\#vehicles \leftarrow min\#veh \leftarrow lowestCost \leftarrow \infty$
 $Ccopy \leftarrow C$
for each $\Delta \in \mathbf{\Delta}$
  while $C \neq \varnothing$ AND $LLimit < \#vehicles$ AND $\#vehicles \leq min\#veh$ do
      $W \leftarrow \min(W, |C|)$
      $C^* \leftarrow \mathrm{w}(start, C, \mathrm{g}, W)$
      for each $v_i \in C^*$
          if $\mathrm{c}(bestSequence \cup v_i) + \mathrm{c}(\mathbf{H}_{\mathrm{r}}(\Delta, v_i, C, v_0)) <$ $lowestCost$ then
              $lowestCost \leftarrow \mathrm{c}(bestSequence \cup v_i) + \mathrm{c}(\mathbf{H}_{\mathrm{r}}(\Delta, v_i, C, v_0))$
              $lowestNext \leftarrow v_i$
          end if
      end for
      $start \leftarrow lowestNext$
      $C \leftarrow C \setminus lowestNext$
      $bestSequence \leftarrow bestSequence \cup lowestNext$
      $R \leftarrow bestSequence \cup \mathbf{H}_{\mathrm{r}}(\Delta, lowestNext, C, v_0)$
      $\#vehicles \leftarrow$ cardinality of the set of routes $R$
  end while
  $C \leftarrow Ccopy$
  if $min\#veh > \#vehicles$
       $min\#veh \leftarrow \#vehicles$
  end if
end for

Output:
  Best set of routes $R$ that serve all $C$ customers
END $\mathbf{H}_{\mathrm{c}}$
```

The conditions in the while-loop that starts in line 7 (i.e., **while** $C \neq \varnothing$ **AND** …) reduce the number of unnecessary computations after a lower bound has been reached or when a particular instance of the cost parameters $\Delta \in \mathbf{\Delta}$ are producing a solution with a larger number of routes. The generalized cost function g that is used in $\mathbf{H}_{\mathrm{r}}$ must not be confused with the objective cost function c that is used in $\mathbf{H}_c$ or the improvement heuristic $\mathbf{H}_{\mathrm{i}}$; the latter cost function is the sum of the accrued vehicle, distance, time or penalty costs, as indicated in the objective function.

3.2.3 The Route-Improvement Algorithm

After the construction is finished, routing costs can be reduced using a route-improvement algorithm. The improvement algorithm works on a subset of routes S. In this algorithm two functions are introduced. The function $\mathrm{k}_p(r_i, S, p)$ returns a set of p routes that belong to S and are located in the proximity of route r_i. In this research, the distance between routes' centers of gravity was used as a measure of geographic proximity. By definition, the distance of route r_i to itself is zero. Hence, the route r_i is always included in the output of the set function $\mathrm{k}_p(r_i, S, p)$. The function $\mathrm{k}_s(R, s)$ orders the set of routes R from smallest to largest based on the number of customers per route and then returns a set of $s \geq 1$ routes with the least number of customers (e.g., $\mathrm{k}_s(R, 1)$ will return the route with the least number of customers). If two or more routes have the same number of customers, ties are solved drawing random numbers. To simplify notation the term $C(S)$ is the set of customers served by the set of routes S.

Functions or Algorithms:

```
        $\mathbf{H}_c$: Route building heuristic, $\mathrm{k}_s$ and $\mathrm{k}_g$: route selection functions
Data:
        $W$ : Number of solutions to be built and compared in the construction heuristic
        $\Delta$ : Generalized cost parameters of the auxiliary route heuristic
        $s$ : Number of routes potentially considered for improvement
        $p$ : Number of neighboring routes to $r_i$ that are reconstructed
        $R$ : Set of routes
        $LLimit$ = lowest number of vehicles or stop condition for the $\mathbf{H}_c$ heuristic
START $\mathbf{H}_i$
$s \leftarrow \min(s, |R|-1)$
$p \leftarrow \min(s, p)$
$S \leftarrow \mathrm{k}_s(R, s) \subseteq R$
$S' \leftarrow R \setminus S$
while $|S|>1$ do
        $r^* \leftarrow \mathrm{k}_s(S, 1)$
        $G \leftarrow \mathrm{k}_p(r^*, S, p)$
        $G' \leftarrow \mathbf{H}_c(\mathbf{H}_r, W, \Delta, s, p, C(G), LLimit)$
if $\mathrm{c}(G') < \mathrm{c}(G)$ then
        $R \leftarrow R \setminus G$
        $R \leftarrow R \cup G'$
        $S \leftarrow S \setminus G$
        $S \leftarrow S \cup G'$
end if
        $r = \mathrm{k}_s(S, 1)$
        $S = S \setminus r$
**if** $|S'| > 0$ **then**
        $r' = \mathrm{k}_s(S', 1)$
        $S \leftarrow S \cup r'$
        $S' \leftarrow S' / r'$
        $s \leftarrow \min(s, |S|)$
        $p \leftarrow \min(s, p)$
  **end while**
Output:
        $R$ set of improved routes
**END** $\mathbf{H}_{\mathrm{i}}$
```

4.0 EXPERIMENTAL SETTING

As seen in the previous section, at its core the IRCI algorithm is a construction algorithm where routes are sequentially built and improved. This section compares the results of the IRCI algorithm against other solution methods that report solution quality and computation time on Salomon benchmark problems for the VRPHTW and VRPSTW. The comparison only includes other construction algorithms or solution approaches that were designed for both hard and soft time windows.

The well-known 56 Solomon benchmark problems for the VRPHTW 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 customers (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.

Table 1. VRPHTW Results for Construction Algorithms vs. IRCI

Average Number of Vehicles by Problem Class

Method	R1	R2	C1	C2	RC1	RC2
(1) Solomon (1987)	13.58	3.27	10.00	3.13	13.50	3.88
(2) Potvin et al. (1993)	13.33	3.09	10.67	3.38	13.38	3.63
(3) Ioannou et al. (2003)	12.67	3.09	10.00	3.13	12.50	3.50
(4) IRCI	12.50	3.09	10.00	3.00	12.00	3.38

Average Distance

Method	R1	R2	C1	C2	RC1	RC2
(1) Solomon (1987)	1,437	1,402	952	693	1,597	1,682
(2) Potvin et al. (1993)	1,509	1,387	1,344	798	1,724	1,651
(3) Ioannou et al. (2003)	1,370	1,310	865	662	1,512	1,483
(4) IRCI	1,262	1,171	872	656	1,420	1,342

Computation time for all 56 problems: (1) DEC 10, 1 run, 0.6 min.; (2) IBM PC, 1 run, 19.6 min.; (3) Intel Pentium 133 MHz, 1 run, 4.0 min. (4) Intel Pentium M 1.6 Mz, 10.9 min

Table 1 presents the summary of the results when construction heuristics for the VRPHTW are compared. Against the three construction heuristics proposed by Solomon, Potvin et al. and Ioannou et al., the IRCI algorithm outperforms them all in classes R1, C2, RC1, and RC2 while tying with the best in classes R2 and C1. Distance-wise, the performance of the IRCI algorithm is superior in all six classes of problems. The IRCI produces results in a relatively short time, less than12 seconds per 100 customer problems on average; however, the other simpler algorithms have shorter running times. The IRCI results presented in Table 1 and 2 were obtained first running a VRPSTW version of the Solomon instances to obtain a set of lower bounds and STW results, and then using those bounds to solve the VRPHTW. The reported time for the IRCI corresponds to the total time to solve both types of problems for all 56 Solomon instances. The other references solve only the VRPHTW type.

Table 2. VRPHTW Results for Metaheuristic Algorithms vs. IRCI

Average Number of Vehicles by Problem Class

Method	R1	R2	C1	C2	RC1	RC2
(1) Taillard et al. (1997)	12.64	3.00	10.00	3.00	12.08	3.38
(2) Ibaraki et al. (2002)	11.92	2.73	10.00	3.00	11.50	3.25
(3) IRCI	12.50	3.09	10.00	3.00	12.00	3.38

Average Distance *by Problem Class*

Method	R1	R2	C1	C2	RC1	RC2
(1) Taillard et al. (1997)	1,220.4	1,013.4	828.5	590.9	1,381.3	1,198.6
(2) Ibaraki et al. (2002)	1,217.4	959.1	828.4	589.9	1,391.0	1,122.8
(3) IRCI	1,261.6	1,170.8	871.8	655.6	1,419.8	1,342.4

Computation time for all 56 problems: (1) Sun Sparc 10, 261 min.; (2) Pentium III 1 GHz, 250 min.; (3) Intel Pentium-M 1.6 Mhz 10.9 min

Table 2 presents the summary of the results when the IRCI algorithm is compared against two metaheuristics presented in the literature review that were explicitly designed to solve both soft and hard time windows: the tabu search heuristics of Taillard et al. (1997) and the composite metaheuristic of Ibaraki et al. (2005). As in Table 1, the reported time for the IRCI corresponds to the total time to solve both types of problems for all 56 Solomon instances. The other references solve only the VRPHTW type.

When compared to the Tabu heuristic of Taillard et al., with its 20 iterations, the results are similar, though the IRCI is faster in computation time even accounting for the different processing speed. The solution method proposed by Ibaraki et al. has a very good solution quality, but at the expense of lengthy computation times.

In the soft time-window benchmark problems, the results of the IRCI are compared against the results of prerequisites: Balakrishnan (1993) – denoted BAL in Tables 3 and 4 – and Chiang and Russell (2004). The latter has two solution methods: tabu search and advance recovery, which are denoted Tables 3 and 4 by the initials TB and AR, respectively.

Balakrishnan (1993) and Chiang and Russell (2004), the only references with time and cost results for a standardized set of problems, solve a subset of Solomon problems setting a $P_{\max}$ that can be either 10, 5, or 0 % of the total route duration $(l_0 - e_0)$. Balakrishnan (1993) and Chiang and Russell (2004) also set a maximum vehicle waiting time limit $W_{\max}$. The maximum waiting time limits the amount of time that a vehicle can wait at a customer location before starting service (i.e., a vehicle can arrive to customer i only after $(e_i - P_{\max} - W_{\max})$). Since the VRPSTW is a relaxation of the VRPHTW, a maximum waiting time constraint $W_{\max}$ is clearly opposed to the spirit of the VRPSTW since a new constraint completely unrelated to time windows is added[2]. Despite these shortfalls, a $W_{\max} = 10\%$ constraint is added, mainly to facilitate comparisons in a level playing field.

Table 3. VRPSTW Results for R1 Problems.

	Wmax	10%				10%			
	Pmax	0%				10%			
	Method	(1) BAL	(2) TS	(3) AR	(4) IRCI	(1) BAL	(2) TS	(3) AR	(4) IRCI
R101	# Veh.	19	19	19	19	15	14	12	13
	Distance	1,915	1,710	1,692	1,639	1,832	1,388	1,212	1,493
	% HTW	100	100	100	100	62	49	8	39
R102	# Veh.	19	17	17	17	14	13	10	12
	Distance	1,890	1,520	1,511	1,481	1,569	1,266	1,173	1,463
	% HTW	100	100	100	100	81	59	33	60
R103	# Veh.		14	13	13	13	11	10	11
	Distance		1,225	1,304	1,284	1,657	1,063	1,013	1,274
	% HTW		100	100	100	83	65	58	73
R109	# Veh.	13	13	12	12	12	11	10	11
	Distance	1,492	1,280	1,165	1,240	1,431	1,102	1,005	1,280
	% HTW	100	100	100	100	90	72	47	82
AVERAGE	# Veh.	17.0	15.8	15.3	15.3	13.5	12.3	10.5	12.3
	Distance	1,766	1,434	1,418	1,411	1,622	1,205	1,101	1,467
	% HTW	100	100	100	100	79.0	61.2	36.5	66.3

Computation time for each STW problem: (1) 25Mhz 80386, 17 to 73 seconds; (2) 2.25 Ghz Athlon, 52 to 82 seconds; (3) 2.25 Ghz Athlon, 448 to 692 seconds; (4) 1.6 Ghz Pentium-M, 4.5 to 4.9 seconds

Table 3 shows the results for the R1 benchmark problems with soft time windows; results for $P_{\max} = 10\%$ and $P_{\max} = 0\%$ are shown. The latter is equivalent to the VRPHTW problem, but with the addition of the $W_{\max} = 10\%$ constraint. In addition to the number of vehicles and distance, Tables 3 and 4 also show the number of customers where the time windows have NOT been relaxed (%HTW); a higher %HTW indicates a better solution quality. As expected, when $P_{\max} = 0\%$ the corresponding % HTW are all equal to 100 because there is no room to relax the customers' time windows.

[2] Further, if there are carrier's costs associated with waiting time (parking), these costs can be incorporated into the routing cost function c rather than imposing a hard-time waiting constraint, which is not usually found in practical problems.

It can be observed that the IRCI algorithms perform very well against Balakrishnan's heuristic. Against tabu search (TS) the IRCI is almost tied, but it performs better in terms of customers that do not have time-window violations. The IRCI solutions are not as good as the advance recovery (AR) method. However, regarding computation times, the IRCI is undoubtedly faster than the TS and without a doubt much faster than the AR method.

Table 4. VRPSTW Results for RC1 Problems.

Wmax		10%				10%			
Pmax		0%				10%			
Method		(1) BAL	(2) TS	(3) AR	(4) IRCI	(1) BAL	(2) TS	(3) AR	(4) IRCI
RC101	# Veh.	16	15	15	15	14	15	11	14
	Distance	2,012	1,719	1,651	1,644	1,795	1,569	1,275	1,839
	% HTW	100	100	100	100	61	62	27	73
RC102	# Veh.	14	13	13	13	13	12	11	13
	Distance	1,808	1,519	1,530	1,575	1,719	1,307	1,222	1,632
	% HTW	100	100	100	100	83	68	56	81
RC103	# Veh.	12	11	11	11	12	10	10	11
	Distance	1,679	1,293	1,284	1,318	1,530	1,228	1,119	1,400
	% HTW	100	100	100	100	92	85	65	92
RC106	# Veh.		12	12	12	13	12	10	12
	Distance		1,445	1,409	1,412	1,620	1,262	1,160	1,487
	% HTW	100	100	100	100	97	77	49	92
AVERAGE	# Veh.	14.0	12.8	12.8	12.8	13.0	12.3	10.5	12.5
	Distance	1,833	1,494	1,469	1,488	1,666	1,342	1,194	1,590
	% HTW	100	100	100	100	83.3	73.0	49.2	84.5

Computation time for each STW problem: (1) 25Mhz 80386, 17 to 73 seconds; (2) 2.25 Ghz Athlon, 52 to 82 seconds; (3) 2.25 Ghz Athlon, 448 to 692 seconds; (4) Intel Pentium-M 1.6 Mhz 4.5 to 4.9 seconds

The same trends are repeated in the RC1 benchmark problems with soft time windows. The IRCI outperforms Balakrishnan's and is competitive with the tabu search (TS) and advance recovery (AR) approach, but at significantly faster running times.

It can be observed that, on average, the IRCI performs well in benchmark instances against simpler and more complex algorithms for hard and soft time windows. The average CPU times are more than reasonable given the relatively modest processing capabilities of a 1.6 Mhz Pentium M laptop. In general, computation times are difficult to compare due to the differences in processing power. Readers who are interested in learning more are referred to Dongarra's work (2007), which includes the results of a set of standard programs to measure processing power and to compare the processing power of different machines. However, comparisons are not straightforward because not all of the processors are included, and there are always differences in codes, compilers, and implementation computational efficiency.

5.0 DISCUSSION

The relative simplicity of the IRCI allows for a straightforward algorithmic analysis. The auxiliary heuristic $\mathbf{H}_{\mathrm{r}}$ is called by the construction algorithm no more than $nW\,|\,\mathbf{\Delta}\,|$ times; where n is the number of customers. Hence, the asymptotic number of operations of the construction algorithm is of order $(nW\,|\,\mathbf{\Delta}\,|\,O(\mathbf{H}_{\mathrm{r}}(n)))$ where $O(\mathbf{H}_{\mathrm{r}}(n))$ denotes the computational complexity of the auxiliary algorithm to route n customers.

The improvement procedure calls the construction procedure a finite number of times. The number of calls is bounded by the number of routes $|R|$. Further, the called computational time of the construction algorithm is $(mW\,|\,\mathbf{\Delta}\,|\,O(\mathbf{H}_{\mathrm{r}}(m)))$ where $m < n$ because only a subset of routes is iteratively improved.

It is clear that the complexity and running time of the auxiliary heuristic $\mathbf{H}_{\mathrm{r}}$ will have a substantial impact on the overall running time. Hence, a generalized nearest neighbor heuristics of (GNNH) is used due to its reduced number of operations and computation time. In particular, if the GNNH has $O(n^2)$ and $W < n$, then the *worst-case complexity* for the IRCI algorithm is of order $O(n^3)$.

To test the *average complexity*, 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 $n = 50$, respectively. Secondly, to create an instance with $n = 200$ customer, for each customer in the original Solomon problem a "clone" is created but with new coordinates. In addition, problem characteristics as clustered, random or random-clustered are retained.

The summary results for the 56 Solomon problems are shown in Table 10. 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 $O(n^2)$ as expected from the complexity analysis and the last column of Table 10.

Table 5. VRPTW Average Run Time Ratios – VRPHTW

(1)	(2)	(3)	(4)	(5)= (4)/(3)*100
n	$O(n^2)$	$O(n^3)$	Run Time Ratio*	% $O(n^3)$
25	1	1	1.0	100%
50	4	8	2.9	36%
100	16	64	15.0	23%
200	64	512	86.3	17%

* The ratio of running times is taking the run time for n=25 as a base.

The proposed IRCI approach can accommodate cost functions that cover most practical applications. The cost functions must be positive functions of fleet size, distance, time or penalties. Cost functions can be asymmetrical (e.g., $\mathrm{p}_e(t) \neq \mathrm{p}_l(t)$ where t accounts for the early or late time). Additionally, cost functions are not required to be linear or identical. Similarly, symmetry is not required and $d_{ij} \neq d_{ji}$ or $t_{ij} \neq t_{ji}$ does not affect the complexity of the algorithm. That is, the corresponding penalty function can be non-convex and discontinuous as long as it is piecewise linear. In addition, customers with two or more time windows can be easily included in the auxiliary route construction algorithm. In addition, the number of routes m is not specified initially and it is an output of the solution algorithm. The bounds for the VRPHTW can be generated endogenously solving a relaxed VRPSTW beforehand.

The relatively simplicity and generality of the IRCI are important factors in real-world applications. Although solution quality and computation times are two key factors to evaluate vehicle routing heuristics, for practical implementations it is also crucial that algorithms are relatively simple and flexible (*Cordeau et al., 2002*). According to Cordeau et al (2002) the majority of the commercial software and in-house routing programs are still based on somewhat simple and unsophisticated methodologies dating back to the 1960s.

Some of the reasons that explain this status quo are (a) dispatchers' preference for algorithms/programs that are highly interactive and allow for manual improvements and the manipulation of constraints and customer priorities; (b) better results on benchmark problems are usually obtained at the expense of too many parameters or complicated coding that lacks flexibility to accommodate real-life constraints; (c) dispatchers may find algorithms with too many parameters difficult to calibrate or even understand; and (d) solution approaches that are markedly tailored to perform well on the benchmark problems may lack generality and robustness in real-life problems. As indicated by Golden et al. (1998), algorithms should also be compared not only by the number of parameters but also by how intuitive and reasonable these parameters are from a user's perspective.

6.0 LITERATURE REVIEW FOR THE TDVRP

Unlike widely studied versions of the VRP (i.e., capacitated VRP or time windows VRP), time-dependent problems have received considerably less attention. The time-dependent VRP was first formulated by Malandraki and Daskin (1989, 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 "restricted" 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 and Daskin (1989, 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.

Fleischmann et al. (2004) utilized 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 nine 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 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.

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 can be swiftly replicated and solution qualities and computation times cannot be compared.

Comparisons are also problematic 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).

Benchmark instances that can be clearly and unmistakably replicated by future researchers are detailed in Section 6.0. The next section introduces mathematical notations and defines the problem under study.

7.0 TDVRP 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 $V=(v_0,....,v_{n+1})$. 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 $q_i \geq 0$, a service time $g_i \geq 0$, 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 $C=\{v_1,....,v_n\}$ 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 $d_{ij} \geq 0$ and a travel time $t_{ij}(b_i) \geq 0$ 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.

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 decision variable y_i^k indicates service start time for customer i served by vehicle k. The TDVRP is formulated as follows:

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

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

subject to:

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

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

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

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

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

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

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

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

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

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

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

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 $e_i \le e_i^{\#}, l_i^{\#} \le l_i$. 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, 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:

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

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

minimize

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

subject to

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

$$x_i^e\,(e_i^{\#} - y_i^k)^+ \leq x_i^e \qquad (18)$$

$$x_i^e \in \{0,1\}\,,\;\; x_i^l \in \{0,1\} \qquad (19)$$

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[3] (14); a tertiary objective is the minimization of early penalties (15); and 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.

[3] Although the cost of early and late service times is 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.).

8.0 TDVRP 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, Braysy and Gendreau, 2005b*). 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 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 presented IRCI solution approach for the TDVRP employs algorithms that do not require modifications to accommodate constant or time-dependent travel speeds. The construction and improvement procedure are *sequential* and originally designed at the *route* level (i.e., it is not a local improvement). Hence, the presented algorithm produces routes for time-dependent vehicle routing problems with hard and soft time windows with similar computation times. This research builds upon previous work to solve the VRP with soft and hard time windows presented in Section 3.0.

8.1 SOLUTION ALGORITHMS

The solution method to minimize fleet size is divided into two phases and algorithms: route construction and route improvement like in the constant-speed case. Travel-time calculations are necessary to execute construction and improvement algorithms. 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, these two additional algorithms are described in this section to calculate travel times and to optimize service times given a set of routes.

8.1.1 Algorithm Used to Calculate Travel Times

Unlike the algorithms presented in Section 8.0, 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 $\mathbf{T} = T_1, T_2, ..., T_p$; each period T_k has an associated constant travel speed s_k. The algorithm is adapted from Ichoua et al. (*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 $k$, $t_{\underline{k}} \le b_i \le t_{\overline{k}}$
$a_j \leftarrow b_i + d_{ij} / s_k$
$d \leftarrow d_{ij}, t \leftarrow b_i$
while $a_j > t_{\overline{k}}$ do
        $d \leftarrow d - (t_{\overline{k}} - t)s_k$
        $t \leftarrow t_{\overline{k}}$
        $a_j \leftarrow t + d / s_{k+1}$
        $k \leftarrow k + 1$
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.

8.1.2 Service Time Improvement

The previous algorithms deal with the minimization of costs via sequencing of customers and their assignment to routes. The $\mathbf{H}_y$ algorithm aims at reducing costs by improving customer service start times for a given set of routes produced by $\mathbf{H}_i$.

For any given route, a dynamic programming approach 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 any given route k defined by the sequence of customers $(0,1,2,...,q,q+1)$ 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 [4]:

$$\pi(y_q, a_q) = c_d d_{q,q+1} + c_t(a_{q+1} - y_q) + c_e(e_q^\# - y_q^k)^+ + c_l(y_q^k - e_l^\#)^+ \tag{20}$$

[4] The distance term can be eliminated from (20) because it is not affected by service time.

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

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 $\overline{y}_i^k$ 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 $[\underline{y}_i^k, \overline{y}_i^k]$ and can be calculated using a forward and backward algorithm.

Proof: 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.

Starting from the depot, latest possible departure time from customer q is:

$\arg\max_{y \in \Re} y,\ s.t.(y + t_{q,q+1}(y) \leq l_{q+1})$; due to the continuous speed function and the FIFO property this value is unique. The latest possible service time at customer q is $\overline{y}_q^k = \min(y - g_q, l_q)$. Latest possible departure time from customer $q-1$ is $\arg\max_{y \in \Re} y,\ s.t.(y + t_{q-1,q}(y) \leq \overline{y}_q^k)$; latest possible service time at customer $q-1$ is $\overline{y}_{q-1}^k = \min(y - g_{q-1}, l_{q-1})$ and so on until reaching the first customer.

Based on the workings of the algorithms $\mathbf{H}_r, \mathbf{H}_c$, and $\mathbf{H}_i$ 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.

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-window 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.

8.1.3 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 $\mathbf{T} = T_1, T_2, ..., T_p$; each period T_k has an associated constant travel speed s_k in the time interval $T_k = [t_{\underline{k}}, t_{\overline{k}}]$.

Data:

$\mathbf{T}$ and S: time intervals and speeds

v_i, v_j, y_j: two customers served in this order in route k, y_j^k is the current service time at customer j

```
START $\mathbf{H}_{yb}$

if $y_j^k < l_j^\# \ \& \ y_j^k < \overline{y}_j^k$ then
    $y_j^k \leftarrow \min(l_j^\#, \overline{y}_j^k)$
end if
find $k$, $t_{\underline{k}} \le y_j^k \le t_{\overline{k}}$
$b_i \leftarrow y_j^k - d_{ji} / s_k$
$d \leftarrow d_{ji}, t \leftarrow y_j^k$
while $b_i < t_{\underline{k}}$ do
```

```
            $d \leftarrow d-(t-t_{\underline{k}})s_k$
            $t \leftarrow t_{\underline{k}}$
            $b_i \leftarrow t-d/s_{k+1}$
            $k \leftarrow k+1$
    **end while**
    $\overline{y}_i^k \leftarrow \min(b_i-g_i, l_i)$
    Output:
            $y_j^k, \overline{y}_i^k$
    **END** $\mathbf{H}_{yb}$
```

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

Data:

$\mathbf{T}$ 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** $\mathbf{H}_{yf}$
**if** $y_i^k > e_j^\#$ & $y_i^k > \underline{y}_i^k$ **then**
        $y_i^k \leftarrow \max(e_i^\#, \underline{y}_i^k)$
**end if**
**find** $k$, $t_{\underline{k}} \le y_i^k \le t_{\overline{k}}$
 $a_j \leftarrow y_i^k + d_{ij}/s_k$
 $d \leftarrow d_{ij}, t \leftarrow y_i^k$
**while** $a_j > t_{\overline{k}}$ **do**
 $d \leftarrow d-(t_{\overline{k}}-t)s_k$
 $t \leftarrow t_{\underline{k}}$
 $a_j \leftarrow t+d/s_{k+1}$
 $k \leftarrow k+1$
**end while**
$\underline{y}_j^k \leftarrow \max(a_j, e_j)$
```

```
Output:
        $y_i^k, \underline{y}_j^k$
**END** $\mathbf{H}_{yf}$
```

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

9.0 TDVRP PROPOSED BENCHMARK PROBLEMS

As mentioned in Section 2.0, 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[5].

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. 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 to 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 2.0, 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.

9.1.1 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.

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

9.1.2 Time-dependent Problems with Hard Time Windows

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:

TD1 = [1.00 , 1.60, 1.05 , 1.60, 1.00],

TD2= [1.00 , 2.00, 1.50 , 2.00, 1.00],

TD3 = [1.00 , 2.50, 1.75 , 2.50, 1.00].

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.

9.1.3 Time-dependent Problems with Soft Time Windows

These instances introduce two congested periods between depot opening and closing times. The depot working time $[e_0, l_0]$ 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*).

10.0 TDVRP 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[6] 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.

As 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. 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 4.0) and the same parameter values. The exact same parameters are used not only in different types of problems (soft vs. hard), but also in different types of instances (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.

10.1.1 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 6. 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

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

computational times. The results reported by Taillard et al. and Donati et al. are average results and computation times over independent runs.

Table 6. VRPTW Results for Classic Solomon Instances – Constant Speed

Average Number of Vehicles by Problem Class

Method	R1	R2	C1	C2	RC1	RC2
(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

Method	R1	R2	C1	C2	RC1	RC2
(1) Best Ever (1987-…)	1,210	952	828	590	1,384	1,119
(2) Taillard et al. (1997)	1,220	1,013	828	591	1,381	1,199
(2) Donati et al. (2008)	1,199	967	828	590	1,374	1,156
(4) IRCI	1,248	1,124	841	626	1,466	1,308

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 7. VRPTW Results – Hard Time Windows

Average Number of Vehicles by Problem Class

Travel time Distribution	R1	R2	C1	C2	RC1	RC2
(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

Travel time Distribution	R1	R2	C1	C2	RC1	RC2
(1) TD1	1,295	1,216	879	657	1,405	1,444
(2) TD2	1,258	1,244	864	654	1,395	1,454
(3) TD3	1,237	1,269	880	697	1,362	1,434

Average Travel Time

Travel time Distribution	R1	R2	C1	C2	RC1	RC2
(1) TD1	1,080	990	729	563	1,164	1,177
(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

The performance of the IRCI algorithm, in relation to other approaches that can solve problems with both soft and hard time windows that have been used in time-dependent problems, is somewhat comparable. The IRCI solutions have relatively low computational times - an average of 21.3 seconds for each 100 customer problems - 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.

10.1.2 Time-dependent Problems with Hard Time Windows

The second set of results corresponds to the Solomon instances with time-dependent travel speeds and soft time windows. In these instances, the primary objective is to minimize the number of vehicles, and 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 Table 7. 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 average 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.

Table 8. VRPTW Results – Soft time windows

Average Number of Vehicles by Problem Class

Travel time Distribution	R1	R2	C1	C2	RC1	RC2
(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

Travel time Distribution	R1	R2	C1	C2	RC1	RC2
(1) TD4	1,142	1,010	856	666	1,241	1,135
(2) TD5	1,131	1,016	860	665	1,226	1,156
(3) TD6	1,127	1,016	869	660	1,236	1,149

Average Travel Time

Travel time Distribution	R1	R2	C1	C2	RC1	RC2
(1) TD4	1,139	1,023	871	669	1,237	1,150
(2) TD5	1,134	1,039	884	672	1,220	1,184
(3) TD6	1,143	1,061	938	685	1,253	1,213

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

10.1.3 Time-dependent Problems with Soft Time Windows

The second set of results corresponds to the Solomon instances with time-dependent travel speeds and soft time windows; results are presented in Table 8. In these instances, the primary objective is to minimize the 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 8 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.

As customary in the VRP with time-windows literature, Table 9 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.

Table 9. VRPTW Results – Soft time windows

Average Number of Soft Time Windows (early)						
Travel time Distribution	R1	R2	C1	C2	RC1	RC2
(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)						
Travel time Distribution	R1	R2	C1	C2	RC1	RC2
(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)						
Travel time Distribution	R1	R2	C1	C2	RC1	RC2
(1) TD4	386.6	1,516.3	516.5	3,025.5	381.7	1,718.9
(2) TD5	425.1	1,609.4	861.5	1,467.7	448.5	1,664.4
(3) TD6	419.3	1,559.1	657.2	1,697.8	446.2	1,508.2
Soft Time Window Penalties (late)						
Travel time Distribution	R1	R2	C1	C2	RC1	RC2
(1) TD4	210.4	681.2	480.5	3,267.1	197.8	695.9
(2) TD5	208.2	637.6	547.6	1,797.7	173.2	692.1
(3) TD6	187.6	629.5	363.2	1,708.8	189.2	787.8

11.0 TDVRP COMPUTATIONAL COMPLEXITY

The relative simplicity of the IRCI allows for a straightforward algorithmic analysis. The auxiliary heuristic $\mathbf{H}_{\mathrm{r}}$ is called by the construction algorithm no more than $nW|\mathbf{\Delta}|$ times; where n is the number of customers. Hence, the asymptotic number of operations of the construction algorithm is of order $(nW|\mathbf{\Delta}|O(\mathbf{H}_{\mathrm{r}}(n)))$ where $O(\mathbf{H}_{\mathrm{r}}(n))$ denotes the computational complexity of the auxiliary algorithm to route n customers. Hence, the complexity and running time of the auxiliary heuristic $\mathbf{H}_{\mathrm{r}}$ 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 $(n_i W|\mathbf{\Delta}|O(\mathbf{H}_{\mathrm{r}}(n_i)))$.

The complexity of the $\mathbf{H}_i$ algorithm is then of order $O(m n_i W|\mathbf{\Delta}|O(\mathbf{H}_{\mathrm{r}}(n_i)))$ 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.

To test the increase in computational running time, instances with different numbers of customers are run. First, 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 an instance with $n=200$ customer, for each customer in the original Solomon problem a "clone" is created but with new coordinates. Problem characteristics as clustered, random or random-clustered are retained.

The summary results for each problem size are shown in Table 10. 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.

Table 10. VRPTW Average Run Time Ratios – TD3

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

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 worse 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 10.

12.0 TDVRP CONCLUSIONS

Readily replicable time-dependent instances with 100 customers were presented and solved with a new route-construction and improvement algorithm. This is the first research effort to publish solutions to time-dependent problems with hard time windows using standard and replicable instances. The computational results indicate that the proposed IRCI algorithms can solve soft and hard time-window, time-dependent vehicle routing problems in relatively small computation times. Furthermore, 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 solution quality of the new algorithm appears to be comparable to other approaches that can be used to solve constant speed and soft time-window problems with time-dependent speeds. However, the proposed IRCI approach seems to have an advantage in TDVRP with hard time windows; problems that cannot be readily tackled by local search heuristics and have not yet been studied in the literature.

The relatively low computational complexity, simplicity and generality of the IRCI are important factors in real-world applications with constant and time-dependent travel times. The algorithms are relatively simple and flexible and their parameters are intuitive. This is a substantial benefit in practical implementations. Two different methods were proposed to group routes and future research efforts may explore alternative grouping methodologies as well as route-construction approaches.

13.0 REFERENCES

AHN, B. H. & SHIN, J. Y. (1991) Vehicle-routing with time windows and time-varying congestion. *Journal of the Operational Research Society,* 42**,** 393-400.

BALAKRISHNAN, N. (1993) Simple Heuristics for the Vehicle Routeing Problem with Soft Time Windows. *The Journal of the Operational Research Society,* 44**,** 279-287.

BERGER, J., BARKAOUI, M. & BRAYSY, O. (2003) A route-directed hybrid genetic approach for the vehicle routing problem with time windows. *INFOR,* 41**,** 179–194.

BRAYSY, I. & GENDREAU, M. (2005a) Vehicle routing problem with time windows, part 1: Route construction and local search algorithms. *Transportation Science,* 39**,** 104-118.

BRAYSY, I. & GENDREAU, M. (2005b) Vehicle routing problem with time windows, part II: Metaheuristics. *Transportation Science,* 39**,** 119-139.

BRAYSY, O. (2002) Fast local searches for the vehicle routing problem with time windows. *Infor,* 40**,** 319-330.

BRAYSY, O. (2003) A Reactive Variable Neighborhood Search for the Vehicle-Routing Problem with Time Windows. *INFORMS Journal on Computing,* 15**,** 347-368.

CALVETE, H. I., GALÉ, C., OLIVEROS, M. J. & SÁNCHEZ-VALVERDE, B. (2007) A goal programming approach to vehicle routing problems with soft time windows star, open. *European Journal of Operational Research,* 177**,** 1720-1733.

CASEAU, Y. & LABURTHE, F. (1999) Heuristics for Large Constrained Vehicle Routing Problems. *Journal of Heuristics,* 5**,** 281-303.

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**,** 1298-1310.

COPOT (2005) City Of Portland Freight Master Plan: Staff Recommendation To Planning Commission. CITY OF PORTLAND OFFICE OF TRANSPORTATION, Portland, OR., September, 2005.

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**,** 512-522.

CORDONE, R. & CALVO, R. W. (2001) A Heuristic for the Vehicle Routing Problem with Time Windows. *Journal of Heuristics,* 7**,** 107-129.

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.

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**,** 1174-1191.

DONGARRA, J. J. (2007) Performance of various computers using standard linear equations software. *Technical Report CS-89-85 - University of Tennessee Nov 20, 2007,* Accessed Nov 23, 2007, http://www.netlib.org/utk/people/JackDongarra/papers.htm.

FERLAND, J. A. & FORTIN, L. (1989) Vehicles Scheduling with Sliding Time Windows. *European Journal of Operational Research,* 38**,** 213-226.

FIGLIOZZI, M. A. (2008) An Iterative Route Construction and Improvement Algorithm for the Vehicle Routing Problem with Soft and Hard Time Windows. *Applications of Advanced*

Technologies in Transportation (AATT) 2008 Conference Proceedings. Athens, Greece, May 2008 - Submitted for publication, currently under 2nd review.

FIGLIOZZI, M. A. (2009) The Impact of Congestion on Commercial Vehicle Tours and Costs. *Forhcoming Transportation Research Part E: Logistics and Transportation*.

FLEISCHMANN, B., GIETZ, M. & GNUTZMANN, S. (2004) Time-varying travel times in vehicle routing. *Transportation Science,* 38**,** 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.* Boston, Kluwer.

HAGHANI, A. & JUNG, S. (2005) A dynamic vehicle routing problem with time-dependent travel times. *Computers & Operations Research,* 32**,** 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**,** 343-351.

HOMBERGER, J. & GEHRING, H. (1999) Two evolutionary metaheuristics for the vehicle routing problem with time windows. *INFOR,* 37**,** 297–318.

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**,** 206-232.

ICHOUA, S., GENDREAU, M. & POTVIN, J. Y. (2003) Vehicle dispatching with time-dependent travel times. *European Journal Of Operational Research,* 144**,** 379-396.

IOANNOU, G., KRITIKOS, M. & PRASTACOS, G. (2001) A greedy look-ahead heuristic for the vehicle routing problem with time windows. *Journal Of The Operational Research Society,* 52**,** 523-537.

IOANNOU, G., KRITIKOS, M. & PRASTACOS, G. (2003) A problem generator-solver heuristic for vehicle routing with soft time windows. *Omega,* 31**,** 41-53.

JUNG, S. & HAGHANI, A. (2001) Genetic Algorithm for the Time-Dependent Vehicle Routing Problem. *Transportation Research Record,* 1771**,** 164-171.

KOSKOSIDIS, Y. A., POWELL, W. B. & SOLOMON, M. M. (1992) An optimization-based heuristic for vehicle routing and scheduling with soft time window constraints. *Transportation science,* 26**,** 69-85.

LIU, F. H. F. & SHEN, S. Y. (1999) A route-neighborhood-based metaheuristic for vehicle routing problem with time windows. *European Journal of Operational Research,* 118**,** 485-504.

MALANDRAKI, C. (1989) Time dependent vehicle routing problems: Formulations, solution algorithms and computational experiments. Evanston, Illinois., Ph.D. Dissertation, Northwestern University.

MALANDRAKI, C. & DASKIN, M. S. (1992) Time-Dependent Vehicle-Routing Problems - Formulations, Properties And Heuristic Algorithms. *Transportation Science,* 26**,** 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**,** 45-55.

POTVIN, J. & ROUSSEAU, J. (1993) A parallel route building algorithm for the vehicle routing and scheduling problem with time windows. *European Journal of Operational Research,* 66**,** 331–340.

RUSSELL, R. A. (1995) Hybrid heuristics for the vehicle routing problem with time windows. *Transportation science,* 29**,** 156-166.

SEXTON, T. R. & CHOI, Y. M. (1986) Pickup and delivery of partial loads with" soft" time windows. *American Journal of Mathematical and Management Sciences,* 6**,** 369-398.

SINTEF (2008) Benchmarks - Vehicle Routing and Travelling Salesperson Problems. SINTEF Applied Mathematics, Department of Optimization, Norway, http://www.top.sintef.no/vrp/benchmarks.html.

SOLOMON, M. M. (1987) Algorithms For The Vehicle-Routing And Scheduling Problems With Time Window Constraints. *Operations Research,* 35**,** 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**,** 170-186.

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**,** 990-1007.

OTREC
OREGON TRANSPORTATION RESEARCH AND EDUCATION CONSORTIUM

P.O. Box 751
Portland, OR 97207

OTREC is dedicated to stimulating and conducting collaborative multi-disciplinary research on multi-modal surface transportation issues, educating a diverse array of current practitioners and future leaders in the transportation field, and encouraging implementation of relevant research results.