Available online at www.sciencedirect.com
SCIENCE DIRECT®

Operations Research Letters 34 (2006) 111 – 120

Operations Research Letters
www.elsevier.com/locate/orl

The multiple TSP with time windows: vehicle bounds based on precedence graphs

Snežana Mitrović-Minić*, Ramesh Krishnamurti[1]

School of Computing Science, Simon Fraser University, Burnaby, BC, Canada V5A 1S6

Received 10 September 2003; accepted 26 January 2005
Available online 21 April 2005

Abstract

Vehicle bounds for the multiple traveling salesman problem with time windows are found by covering two precedence graphs with the minimum number of paths. Instances with tight bounds are presented, as well as instances for which the bounds are loose. The similarity of these instances is discussed.
© 2005 Elsevier B.V. All rights reserved.

Keywords: Multiple traveling salesman problem with time windows; Bounds for the minimum number of vehicles

1. Introduction

The *multiple traveling salesman problem with time windows* (m-TSPTW) deals with finding optimal routes for a fleet of vehicles in order to serve a set of locations, each within a specified time window. This article describes an algorithmic graph theory approach to finding a lower and an upper bound for the minimum number of vehicles needed to serve all the locations. The study has been motivated by the fact that the m-TSPTW is an NP-hard problem, and that a polynomial-time procedure for evaluating the number of vehicles required may be a useful preprocessing step when solving the m-TSPTW. Furthermore, our work represents an extension of [5].

1.1. Literature review

The m-TSPTW is a generalization of the *traveling salesman problem* (TSP), a well-known NP-hard problem. Checking feasibility of the m-TSPTW—when the number of vehicles is fixed—is an NP-complete problem [11]. Therefore, the problem of finding the minimum number of vehicles v^* needed to serve the set of locations of m-TSPTW is NP-hard.

The m-TSPTW belongs to the class of time-constrained vehicle routing problems which has been extensively studied in the last few decades. For

* Corresponding author. Tel.: +1 604 291 5938; fax: +1 604 291 3045.
E-mail address: smitrovi@cs.sfu.ca (S. Mitrović-Minić).

[1] This research was partly supported by the Canadian Natural Sciences and Engineering Research Council. The support is gratefully acknowledged. Thanks are due to referees for their valuable comments.

extensive surveys and bibliography, see the state-of-the-art books [2,14].

Our work may be viewed as an extension of [5] where the chain merger problem is solved by means of a generalized Dilworth's theorem. The solution procedure is an incremental chain decomposition procedure starting from an initial decomposition. The generalized Dilworth's theorem is used over a graph where each vertex represents a chain, and each edge indicates feasibility of merging two chains. The aim was to define a neighborhood for a tabu search procedure applied to the pickup and delivery problem with time windows [6].

1.2. Problem formulation

Let $G=(V,A)$ be a complete graph, where $V=\{1,2,3,\ldots,n\}$ is a set of locations and A is a set of arcs. A travel time $t_{i,j}>0$ is associated with every arc $(i,j)\in A$, $i\neq j$, such that the triangle inequality is satisfied. Each location $i\in V$ has a time window $I_i=[a_i,b_i]$, where a_i is the release time and b_i is the deadline. Service at each location has to start before its deadline. Service time at each location is zero, although a non-zero service time s_i at location i may be handled by increasing each travel time from i by s_i. If a vehicle arrives too early at a given location, it is allowed to wait. The m-TSPTW consists of determining optimal routes for a fleet of vehicles in order to serve each location in V exactly once, satisfying time window constraints.

This article presents two types of precedence graphs used for finding a lower bound $\underline{v^*}$ and an upper bound $\overline{v^*}$ for the minimum number of vehicles v^* needed to serve locations of the m-TSPTW. The remainder of the article is organized as follows. Section 2 introduces the two precedence graphs and describes an algorithm for finding bounds. Section 3 describes classes of m-TSPTW instances for which the two bounds are tight, and classes of instances for which one of the bounds is loose. Section 4 presents results of an experimental study. Section 5 concludes the paper. Proofs for the lemmas of Section 3 are given in the appendix.

2. Precedence graphs and vehicle bounds

This section introduces two precedence graphs and presents algorithms for finding a lower and an upper bound for the minimum number of vehicles needed to serve locations for the m-TSPTW [9].

2.1. Precedence graphs

A *precedence graph* is a directed graph which mirrors the precedence relation between pairs of locations in V, imposed by the travel times and the time window constraints. We present two types of precedence graphs, the start-time precedence graph and the end-time precedence graph. The start-time precedence graph is known in the literature of time-constrained vehicle routing problems as the admissible graph [2].

Definition 1. The start-time precedence graph, $G_a=(V,A_a)$, is a directed graph (digraph) of the precedence relations among locations in V with regard to their release times. Set A_a of arcs is given by

$$A_a=\{(i,j)\in(V\times V)\mid a_i+t_{i,j}\leqslant b_j\}.$$

Definition 2. The end-time precedence graph, $G_b=(V,A_b)$, is a digraph of the precedence relations among locations in V with regard to their deadlines. Set A_b of arcs is given by

$$A_b=\{(i,j)\in(V\times V)\mid b_i+t_{i,j}\leqslant b_j\}.$$

It is clear that $A_b\subset A_a$, implying that graph G_b is a subgraph of graph G_a.

Fig. 1 shows an m-TSPTW instance with its precedence graphs. The instance has five locations for which the time windows are $I_1=[0,1]$, $I_2=[1,2]$, $I_3=[3,4]$, $I_4=[0,4]$, $I_5=[2,4]$. The travel times are associated with the edges of G.

When dealing with an m-TSPTW instance with a depot, the precedence graphs do not have to contain the depot. Before building the precedence graphs, set the release time a_i of each location i to $\max\{a_i,a_0+t_{0,i}\}$, where 0 is the depot and a_0 is the earliest departure time of a vehicle from the depot. If vehicles have to return to the depot before a specified time b_0, set the deadline b_i of each location i to $\min\{b_i,b_0-t_{i,0}\}$.

When all arcs $(k,i)\in A_b$ incoming to location i are such that $b_k+t_{k,i}<b_i$, the deadline b_i may be reduced to $\min\{b_i,\max_k(b_k+t_{k,i})\}$.

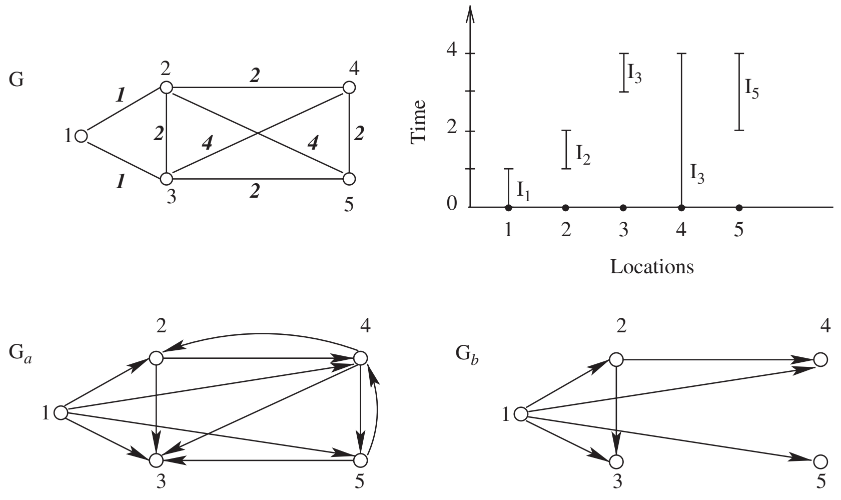

Fig. 1. An m-TSPTW instance with its precedence graphs.

2.2. Path in a precedence graph

The following two lemmas discuss the number of vehicles required for serving locations on a path in the precedence graphs.

Lemma 1. *For serving locations on a path P of graph G_a more than one vehicle may be needed.*

Proof. Consider three consecutive locations $(i, j, l) \subset P$, and assume that $a_i + t_{i,j} > a_j$, If inequality $a_i + t_{i,j} + t_{j,l} > b_l$ holds, then for serving locations (i, j, l) at least two vehicles are needed. □

Lemma 2. *All locations on a path P of graph G_b may be served by one vehicle.*

Proof. Follows from the definition of the end-time precedence graph. □

2.3. A lower and an upper bound

The two precedence graphs of an m-TSPTW instance may be used for finding a lower bound and an upper bound for the minimum number of vehicles needed. The following two theorems are corollaries of Lemmas 1 and 2, respectively.

Theorem 1. *The minimum number of directed paths that cover all the nodes of the start-time precedence graph G_a is a lower bound $\underline{v^*}$ on the minimum number of vehicles v^* needed to serve all locations in G.*

Theorem 2. *The minimum number of directed paths that cover all the nodes of the end-time precedence graph G_b is an upper bound $\overline{v^*}$ on the minimum number of vehicles v^* needed to serve all locations in G.*

A lower bound for the *vehicle routing problem with time windows* (VRPTW) equal to our lower bound may be found by solving the maximum clique problem over an incompatibility graph—bound LB_2 proposed in [7]. Our approach differs in that the time complexity is polynomial. Another lower bound LB_3 has achieved better results than LB_2 on many Solomon instances, but it may be used only for the problems with the depot and when vehicle routes have to be closed.

The end-time precedence graph G_b is transitive and acyclic. A directed graph is *transitive* if the existence of two arcs (i, j) and (j, k) implies the existence of arc (i, k), for any three nodes $i, j, k \in V$. In order to prove transitivity, assume that $(i, j) \in A_b$ and $(j, k) \in A_b$, i.e., $b_i + t_{i,j} \leqslant b_j$ and $b_j + t_{j,k} \leqslant b_k$. Due to the triangle inequality, the following holds: $b_i + t_{i,k} \leqslant b_i + t_{i,j} + t_{j,k} \leqslant b_j + t_{j,k} \leqslant b_k$, implying that $(i, k) \in A_b$, and that G_b is transitive. Assume now that

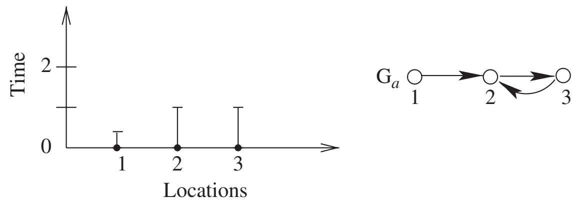

Fig. 2. An m-TSPTW instance for which the start-time precedence graph is not transitive nor acyclic.

G_b contains cycle $C=(i_1,i_2,\ldots,i_k)$. Due to positive travel times, $b_{i_1}<b_{i_2}<\cdots<b_{i_k}<b_{i_1}$ holds, which is a contradiction. Therefore, G_b is acyclic.

The start-time precedence graph G_a may have a cycle. An example with three collinear locations is given in Fig. 2. Travel times are $t_{1,2}=1$, $t_{2,3}=t_{3,2}=1$, $t_{1,3}=2$. In such a case, condensation is applied on G_a. The *condensation* of digraph $D=(V,A)$ results in the acyclic digraph $D^{\mathrm{c}}=(V^{\mathrm{c}},A^{\mathrm{c}})$ where $V^{\mathrm{c}}=\{C_1,\ldots,C_{\mathrm{c}}\}$ is a set of strongly connected components of D and $A^{\mathrm{c}}=\{(C_u,C_v),u\neq v \mid \exists i\in C_u \text{ and } \exists j\in C_v \text{ such that } (i,j)\in A\}$. The condensation may be done in $\Theta(|V|+|A|)$ time [10]. In [5], different procedure is used for removing cycles.

The start-time precedence graph G_a is not necessarily transitive. An example is given in Fig. 2. In such a case, the transitive closure of G_a is used. The *transitive closure* of a directed graph $G=(V,A)$ is a graph $G^+=(V,A^+)$ such that for all $i,j\in V$ there is an edge $(i,j)\in A^+$ if and only if there is a directed path from i to j in G. The time needed to generate the transitive closure of a digraph is $\mathrm{O}(|V||A|)$ [10].

An acyclic transitive directed graph uniquely defines a partial order relation among its nodes [15]. In this case, finding a minimum cover of the nodes by directed paths is equivalent to decomposition by chains—a problem from the theory of partially ordered sets. Dilworth's theorem [3] states that the minimum number of chains (totally ordered sets) which partitions a partially ordered set is equal to the maximum size of an antichain (set of incomparable elements). Decomposition by chains of a directed graph $G=(V,A)$ may be solved in polynomial time [4] by solving the maximum bipartite matching problem for graph $B=(V,V;E)$. The set of edges E contains edge $\{i,j\}$ if and only if $(i,j)\in A$. After solving the maximum bipartite matching, the chains are constructed from the matching.

We give below a polynomial-time algorithm for finding a lower bound $\underline{v^*}$.

```
Input:   An m-TSPTW instance.
Output:  The number of chains is a lower bound on
         the minimum number of vehicles needed to
         serve all locations of the m-TSPTW
         instance.
begin
         create precedence graph $G_a$
         $G_a \leftarrow$ condensation of $G_a$
         $G_a \leftarrow$ transitive closure of $G_a$
         solve the decomposition by chains problem
         over $G_a$ by solving the corresponding maxi-
         mum bipartite matching problem
end
```

An algorithm for finding an upper bound $\overline{v^*}$ is similar, except that G_b is used instead of G_a. The steps involving condensation and transitive closure are not required.

The complexity of the algorithm for finding a lower bound is $\mathrm{O}(|V||A|)$ due to the time needed for generating the transitive closure. The complexity of finding an upper bound is $\mathrm{O}(\sqrt{|V|}|A|)$ due to the complexity of the maximum bipartite matching problem [1].

3. Classes of m-TSPTW instances

We now describe how tight these bounds are by presenting several classes of m-TSPTW instances. Some classes guarantee tight bounds. For the other classes, one of the bounds becomes loose. The existence of arbitrarily bad instances is not surprising due to the interrelation of time and space aspects.

3.1. *Similarity of an instance with tight bounds with an instance with a loose bound*

Consider an instance with n locations and with the following characteristics: locations are ordered along a straight line, the travel time $t_{i,i+1}$ between adjacent locations is a constant $c>0$, the travel time $t_{i,j}$ is $c|j-i|$, the width of each time window is equal to c, and $a_{i+1}=a_i+c$, $a_j=a_i+c(j-i)$. The lower bound, as well as the upper bound, equals one (see Fig. 3).

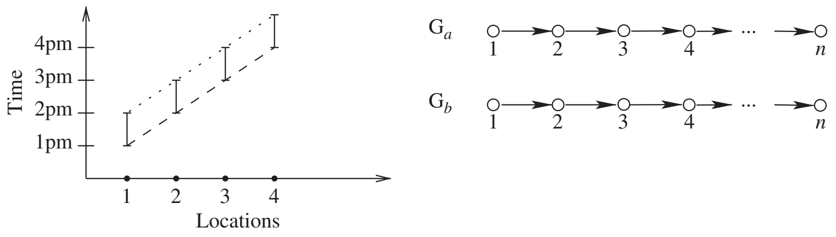

Fig. 3. The m-TSPTW instance containing n collinear locations for which $\underline{v^*} = 1 = v^* = \overline{v^*}$. The travel time between each pair of neighboring locations is 1 h. The width of each time window is 1 h. The dashed line is used for generating G_a, and the dotted line is used for generating G_b.

Consider an instance differing from the above in the position of time windows—all but the first time window are moved down slightly. The time window of location i is moved down with respect to the previous location, i.e., $a_i = a_{i-1} + c - \delta$, where $0 < \delta < c$. In this instance the lower bound stays the same, while the upper bound equals n (see Fig. 4). The minimum number of vehicles required is one.

Lemma 3. *Assume that an m-TSPTW instance with n locations satisfies the following: locations are ordered along a straight line, the travel time $t_{i,j}$ is $c|j-i|$, where c is a positive constant, the width of each time window is equal to c, and $a_j = a_i + (c-\delta)(j-i)$, $\delta > 0$. If $\delta \leqslant c/(n-1)$, then $\underline{v^*} = 1$ and $\overline{v^*} = n$.*

3.2. Simple classes of instances with one loose bound

Fig. 5 shows an instance for which the lower bound is arbitrarily bad, i.e., $v^* - \underline{v^*} = \Omega(n)$, while Fig. 6 depicts another instance for which the upper bound is arbitrarily bad, i.e., $\overline{v^*} - v^* = \Omega(n)$. Therefore, even in simple structured instances with identical time windows, the bounds may be loose. This is not unexpected. The space and time aspects of the problem permit arbitrarily bad instances. See [12] where instances with a worst-case performance ratio of $\Omega(n)$ are provided for a variety of m-TSPTW heuristics.

3.3. Classes of instances with equal bounds

This section presents classes of m-TSPTW instances for which the lower bound is equal to the upper bound. We have studied instances for which $G_a = G_b$. Since $A_b \subset A_a$ it is enough to find the instances in which $A_a \subset A_b \Leftrightarrow (a_i + t_{i,j} \leqslant b_j) \Rightarrow (b_i + t_{i,j} \leqslant b_j),\ \forall (i,j)$.

Lemma 4. *Consider an m-TSPTW instance for which the time windows are of constant width, $b_i - a_i = c$, for a positive constant c. The two precedence graphs are equal, i.e., $G_a = G_b$, if for each location pair (i, j), either $a_i + t_{i,j} \leqslant a_j$ or $a_i + t_{i,j} > b_j$.*

Instances with constant time window widths that do not belong to the class described by Lemma 4 are those for which $\exists (i, j)$ such that $a_j < a_i + t_{i,j} \leqslant b_j$, i.e., for which a vehicle leaving location i at a_i reaches location j within its time window, I_j. These instances are characterized by $A_a \supset A_b$, and their upper bound and lower bound may differ.

One subclass of the class specified by Lemma 4 is described below: Consider an instance in which the travel time between any pair of locations is a multiple of a constant, i.e., $t_{i,j} = p_{i,j}c$, where $p_{i,j}$ is a positive integer, the size of each time window equals c, each time window starts at a multiple of c, i.e., $a_i = q_i c$, where q_i is a positive integer, and $a_i + t_{i,j} \neq b_j,\ \forall i, j \in V$. Note that the last condition may be removed if the time window width is changed to $c - \varepsilon$, where $\varepsilon \in (0, c]$.

We conclude that when the time windows are of equal widths and small enough such that the travel times may be specified as an integer multiple of the time windows width, situations arise in which the two precedence graphs are equal. The bounds then coincide and are equal to the minimum number of vehicles needed to serve all the locations. For example, if all time windows are of width $c = 15 - 1 = 14$ min, all time windows start at a multiple of 15, and the travel

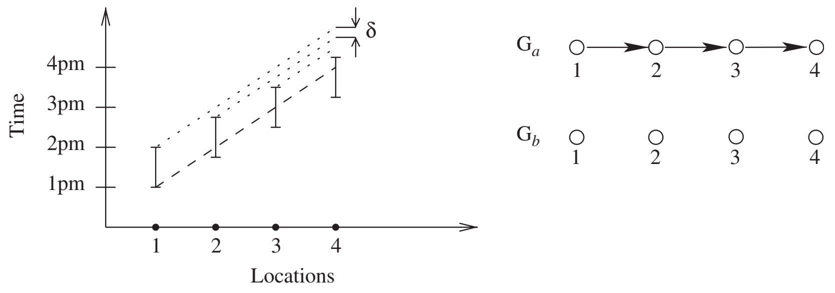

Fig. 4. The m-TSPTW instance containing four locations for which $\underline{v^*}=1=v^*<\overline{v^*}=4$. The travel time between each pair of neighboring locations is 1 h. The width of each time window is 1 h. When $\delta \leqslant 1/n-1$, the instance with n locations will have $\underline{v^*}=1=v^* \ll \overline{v^*}=n$.

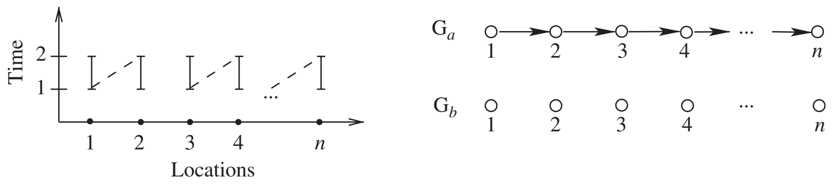

Fig. 5. The m-TSPTW instance containing n locations for which $\underline{v^*}=1 \ll v^*=n/2<\overline{v^*}=n$. The travel time between each pair of neighboring locations is 1. The width of each time window is 1. The dashed lines show the optimal solution.

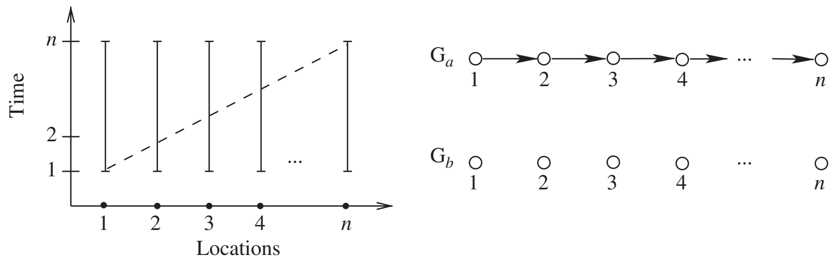

Fig. 6. The m-TSPTW instance containing n locations for which $\underline{v^*}=1=v^*=1 \ll \overline{v^*}=n$. The travel time between each pair of neighboring locations is 1. The width of each time window is $n-1$.

times are multiples of 15, i.e., $t_{i,j} \in \{15, 30, 45, \ldots\}$, the lower and the upper bounds coincide.

Instances with these characteristics may arise when solving the *dial-a-ride problem* (DARP) except that DARP may have an additional constraint that forbids waiting while a vehicle carries a passenger. (Even with this DARP constraint, the lower bound would still be valid.) Another real-life problem with similar instances may be a maritime transportation problem dealing with clearing of typically small loading/unloading areas in a dock yard in order to efficiently use large and expensive cranes. Our approach may give bounds for the number of trucks required to prevent idling of the cranes, if the time windows are of small constant width and if the travel times may be approximated as multiples of the time windows width.

4. Computational study

This section reports the results of an empirical study performed on the original set of Solomon's instances for the VRPTW [13]. This benchmark contains 56 instances divided into six different subsets: C1, C2, R1, R2, RC1 and RC2. Each subset contains between eight and twelve instances. The subset names have the following meaning: C instances have clustered locations for which the time windows were generated based on a known solution; R instances have locations generated uniformly randomly over a square; RC instances have a combination of randomly placed and clustered locations. Instances of Type 2 have wider time windows than the instances of Type 1. There are three problem sizes containing 25, 50 or 100 locations, with 56 instances for each problem size. For each instance, the service period is the same for all three problem sizes.

We have transformed each Solomon's VRPTW instance to an m-TSPTW instance by discarding the vehicle capacity and customer load information. Since each instance has a depot, the preprocessing step described at the end of Section 2.1 is required. The depot time window $[a_0, b_0]$ determines the problem service period $s = b_0 - a_0$.

The results are presented in Table 1. For each problem instance a lower and an upper bound is determined using precedence graphs. The gap (difference) between bounds is found and normalized over the problem size, i.e., the number of locations. This value is multiplied by 100, and thus the gap is represented as a percentage of the problem size. Then, an average for each instance is calculated over the three problem sizes. This value is reported in the table along with its corresponding standard deviation.

Table 2 gives statistics for time windows for the instances. For each Solomon's instance the average value and the standard deviation for time window widths are found. These two values are normalized over the service period length. The values are multiplied by 100 so that the average value and the standard deviation are represented as a percentage of the service period. Then, the average of both values is calculated over the three problem sizes and reported.

Table 2 is provided to ascertain if there is any correlation between the difference in the bounds and the width of time windows in the Solomon's instances. Narrow time windows together with low standard deviation may indicate close bounds. Examples are C101, C106, C201, C205, C206, R101, R201 and RC201 with the gap between the bounds below 12%. Their time window widths have average values between 4.35% and 14.08%, and standard deviation between 0% and 5.01%. But as expected, narrow and uniform time windows do not guarantee close bounds. A counterexample is RC101 for which the time windows are similar to those of RC201, but the gaps between the bounds are significantly different:

Table 1
Gap between lower and upper bound presented as a percentage of problem size

	C1		C2		R1		R2		RC1		RC2	
	Avg	StDev	Avg	StDev	Avg	StDev	Avg	StDev	Avg	StDev	Avg	StDev
1	**10.33**	**1.53**	**5.33**	**2.31**	**9.67**	**2.08**	**11.00**	**4.58**	**25.33**	**1.15**	**11.67**	**1.53**
2	18.00	2.00	26.33	2.08	25.33	4.16	30.33	2.08	**33.33**	**2.31**	28.67	1.15
3	48.00	0.00	50.33	2.08	52.00	2.00	54.67	2.31	54.00	3.46	54.67	2.31
4	69.67	8.50	74.33	6.03	74.67	6.11	75.33	6.43	77.33	5.03	76.33	4.04
5	**17.33**	**2.31**	**9.33**	**2.31**	**24.67**	**3.06**	**14.33**	**1.53**	**32.67**	**3.06**	**19.33**	**4.16**
6	**10.00**	**6.00**	**7.67**	**0.58**	28.67	3.06	30.33	2.08	**35.67**	**2.52**	**13.67**	**2.08**
7	**21.33**	**2.31**	**11.33**	**1.15**	52.67	3.06	55.00	2.65	**37.33**	**2.31**	**15.33**	**4.16**
8	**26.67**	**1.15**	**16.33**	**3.21**	75.67	6.66	75.67	6.66	**46.33**	**7.09**	**20.67**	**3.06**
9	**32.00**	**3.46**			**42.00**	**6.00**	**13.33**	**1.15**				
10					**37.00**	**8.89**	**14.67**	**1.15**				
11					**40.00**	**0.00**	**18.67**	**1.15**				
12					**46.67**	**5.03**						

Table 2
Average time window width presented as a percentage of the problem service period

	C1		C2		R1		R2		RC1		RC2	
	Avg	StDev	Avg	StDev	Avg	StDev	Avg	StDev	Avg	StDev	Avg	StDev
1	**4.89**	**0.89**	**4.72**	**0.00**	**4.35**	**0.01**	**11.54**	**3.60**	**12.50**	**0.05**	**12.50**	**0.01**
2	27.55	37.45	28.90	40.57	26.10	33.18	34.05	37.35	**30.27**	**23.84**	34.22	35.05
3	48.29	42.39	51.23	46.02	45.29	36.40	54.61	41.95	46.56	27.78	54.27	39.78
4	68.57	36.86	72.96	39.91	64.11	32.38	74.67	36.32	62.69	26.63	73.83	34.66
5	**9.79**	**1.78**	**9.44**	**0.00**	**13.04**	**0.03**	**24.00**	**0.02**	**23.05**	**17.94**	**23.66**	**17.44**
6	**8.74**	**5.01**	**14.08**	**2.56**	32.50	29.33	43.22	31.67	**25.00**	**0.06**	**25.00**	**0.01**
7	**14.56**	**0.02**	**21.09**	**9.30**	49.58	32.10	60.78	35.57	**36.43**	**13.34**	**35.88**	**16.37**
8	**19.58**	**3.55**	**18.88**	**0.01**	66.34	28.80	77.88	30.81	**46.45**	**12.51**	**49.04**	**7.51**
9	**29.13**	**0.02**			**25.53**	**3.86**	**34.44**	**15.72**				
10					**37.13**	**16.38**	**38.64**	**24.25**				
11					**40.91**	**24.51**	**47.08**	**7.20**				
12					**50.99**	**7.45**						

RC101 has a gap of 25.33%, while RC201 has a gap of 11.67%. Also compare RC201 to C206.

There are also instances with wide and diverse time windows for which the bounds are relatively close. Instance R210 has wide time windows (38.64%) with a large standard deviation (24.25%) and has the gap between the bounds equal to 14.67%.

Average travel times and the gap between bounds do not appear to be related. Travel times are equal across one instance type. Given as a percentage of the service period, the average travel times and the standard deviations for instances C1, C2, R1, R2, RC2 and RC2, are respectively (2.15%, 1.12%), (0.91%, 0.45%), (15.69%, 7.10%), (3.61%, 1.63%), (21.88%, 10.93%) and (5.47%, 2.73%).

The tables entries that are not boldface indicate instances containing consistently, over all three problem sizes, more than 23% of time windows that are larger than 80% of the problem service period. Locations with these time windows may almost be considered as locations without time windows constraint, and we will refer to them as 'no-time-window' locations throughout this section. Instances with around 25% of 'no-time-window' locations are C102s, C202s, R102s, R106s, R202s, R206s, RC103s and RC202s. The only exception is RC103 with 50 locations containing 14% of 'no-time-window' locations, but still the average number of 'no-time-window' locations over the three problem sizes is 23.3%. Instances with around 50% of 'no-time-window' locations are C103s, C203s, R103s, R107s, R203s, R207s, RC104s and RC203s. Instances with around 70% of 'no-time-window' locations are C104s, C204s, R104s, R108s, R204s, R208s and RC204s. As expected, our approach does not perform very well for problem instances with 'no-time-window' locations. Almost each 'no-time-window' location would be an isolated node in the end-time precedence graph. Therefore, for each such node one vehicle would be reserved in calculating the upper bound.

Considering only the results reported in boldface, the average value of the gap between the bounds is 22.84% with a standard deviation of 12.39%. Particularly good bounds have been found for the 25-location instances C106, C201, C206, for the 50-location instances C201, C205, C206, R101, and for the 100-location instances C201, C205, C206, R201. For all of these instances, the gap between the bounds is between 4% and 8%. (The separate values for each problem size are not shown in the table.) In summary, the two bounds are respectively (4, 5), (2, 4), (2, 4); (2, 4), (1, 5), (1, 5), (12, 16); (3, 7), (1, 9), (1, 8), (2, 9), where the first number is the lower bound and the second number is the upper bound in the number of vehicles. If the instances with around 25% of 'no-time-windows' locations are included, the average value of the gap between the bounds is 24.94% with a standard deviation of 12.55%. Overall, the average value of the gap between the bounds is 34.62% with a standard deviation of 21.77%.

Based on this computational study, it appears that the bounds would be useful for the Solomon instances that are characterized by narrow time windows with relatively uniform width. Even though in general the bounds may differ widely, the fact that they can be computed in polynomial time may render them useful when solving the m-TSPTW. In addition, the algorithms are relatively simple and easy to implement.

5. Conclusion

The paper proposes polynomial algorithms for finding vehicle bounds for the m-TSPTW. From comparing instances with tight and loose bounds, we conclude that based on the size of time windows alone no assumption may be made as to how good the bounds are. An important factor is the relation between the time windows and the travel times.

Finding a range for the number of vehicles needed may be a useful preprocessing step when solving the m-TSPTW. We also believe that this study may initiate further research towards better understanding why in empirical studies some instances of the traveling salesman problem with time windows have been more difficult to solve than others [8].

Appendix

Proof of Lemma 3. By showing $a_i + t_{i,j} \leqslant b_j, \forall i < j$, we show that $\underline{v^*} = 1$.

$$\begin{aligned} a_i + t_{i,j} &= a_j - (c - \delta)(j - i) + c(j - i) \\ &= a_j + \delta(j - i) \\ &\leqslant a_j + \frac{c}{n-1}(j - i) \leqslant a_j + c = b_j. \end{aligned}$$

Consequently, the lower bound $\underline{v^*} = 1$. On the other hand, the upper bound is $\overline{v^*} = n$, because for every pair (i, j), $i < j$ the inequality $b_i + t_{i,j} > b_j$ is valid.

$$\begin{aligned} b_j &= a_j + c = a_i + (c - \delta)(j - i) + c \\ &= b_i + c(j - i) - \delta(j - i) < b_i + c(j - i) \\ &= b_i + t_{i,j}. \end{aligned}$$

Furthermore $\forall i < j$, $(j, i) \notin A_b$. Therefore, the upper bound $\overline{v^*}$ equals n. □

Proof of Lemma 4. Inequality $a_i + t_{i,j} \leqslant a_j$ implies $(i, j) \in A_a$. Furthermore, $(i, j) \in A_b$ because $b_i + t_{i,j} = a_i + c + t_{i,j} \leqslant a_j + c = b_j$. Inequality $a_i + t_{i,j} > b_j$ implies $(i, j) \notin A_a$. It is also valid that $(i, j) \notin A_b$ because $b_i + t_{i,j} > a_i + t_{i,j} > b_j$. □

References

[1] R.K. Ahuja, T.L. Magnanti, J.B. Orlin, Network Flows, Prentice-Hall, Englewood Cliffs, NJ, 1993.

[2] M.O. Ball, T.L. Magnanti, C.L. Monma, G.L. Nemhauser (Eds.), Network Routing, Handbooks in Operations Research and Management Science, vol. 8, North-Holland, Amsterdam, 1995.

[3] R. Dilworth, A decomposition theorem for partially ordered sets, Ann. Math. 51 (1950) 161–166.

[4] L.R. Ford, D.R. Fulkerson, Flows in Networks, Princeton University Press, Princeton, NJ, 1962.

[5] J.N. Hooker, N.R. Natraj, A generalized Dilworth's theorem, with application to routing and scheduling, Working paper, Graduate School of Industrial Administration, Carnegie Mellon University, Pittsburgh, USA, 1992.

[6] J.N. Hooker, N.R. Natraj, A generalized Dilworth's theorem, with application to routing and scheduling, Transport. Sci. 29 (1995) 30–44.

[7] G. Kontoravdis, J.F. Bard, A GRASP for the vehicle routing problem with time windows, ORSA J. Comput. 7 (1995) 10 –23.

[8] O.B.G. Madsen, The vehicle routing problem with time windows, Spring School on Logistics and Distribution Management, Sponsored by the Canada Research Chair in Distribution Management and MITACS, École des Hautes Études Commerciales, Montréal, 8–10 May 2002.

[9] S. Mitrović-Minić, The dynamic pickup and delivery problem with time windows, Ph.D. Dissertation, School of Computing Science, Simon Fraser University, Burnaby, Canada, 2001.

[10] D.F. Robinson, L.R. Foulds, Digraphs: Theory and Techniques, Gordon and Breach Science Publishers, New York, 1980.

[11] M.W.P. Savelsbergh, Local search in routing problems with time windows, Ann. Oper. Res. 4 (1985) 285–305.

[12] M.M. Solomon, On the worst-case performance of some heuristics for the vehicle routing and scheduling problem with time window constraints, Networks 16 (1986) 161–174.
[13] M.M. Solomon, Algorithms for the vehicle routing and scheduling problem with time window constraints, Oper. Res. 35 (1987) 254–265.
[14] P. Toth, D. Vigo, The Vehicle Routing Problem, SIAM Monographs on Discrete Mathematics and Applications, SIAM, Philadelphia, 2002.
[15] W.T. Trotter, Combinatorics and Partially Ordered Sets, The Johns Hopkins University Press, Baltimore, 1992.