Single-vehicle Scheduling Problem on a Straight Line with Time Window Constraints

Chi-lok Chan and Gilbert H. Young
The Chinese University of Hong Kong

Abstract

The problem of single-vehicle scheduling with ready time and deadline constraints can be applicable to many practical transportation problems. But the general case is a generalization of classical Traveling Salesman Problem and it is NP-hard. Psaraftis[1] presented an $O(n^2)$ algorithm for vehicle scheduling problems where the topological structure of sites are restricted to straight line and there is no deadline for each site. He conjectured that this problem is NP-hard if each site has an arbitrary ready time and deadline (called *time window*). In this paper, we prove the conjecture and show that the decision versions of vehicle scheduling problems on a straight line topology with time window are NP-complete for both path and tour versions. We also give an $O(n^2)$ algorithm to solve a special case of this problem in which all sites have a common ready time. This algorithm is applicable to both path and tour versions.

Keywords: deadline, NP-complete, optimal schedule, ready time, vehicle scheduling problem, time windows

1 Introduction

Single-vehicle scheduling problems (VSP) are of great practical importance. One vehicle is going to visit a set of sites which are connected according to some arbitrary graph structure. The practical goals of scrutinizing these problems are minimization of total time taken for the vehicle to visit all sites. In practice, each site has its earliest (*ready time*) and latest (*deadline*) visiting times. The problem become constrained by time windows so that the vehicle should visit every site after the ready time but no later than the deadline. This is called the *path version* of the VSP. If the vehicle has to return to the starting site at the end of the trip, it is called a *tour version*. The literature on the problems with time windows has been growing at an explosive rate over the last few years [2][3][4]. Unfortunately, many of the interesting routing problems are NP-complete[2]. Karuno proved the VSP on a tree topology with arbitrary degree and ready time only is NP-hard [5]. Psaraftis [1] proposed a new model for routing of ships for a shoreline topology. He gave an $O(n^2)$ algorithm for solving the path version of the VSP on a convex shoreline with ready time only, where n denotes the number of sites. This topology is equivalent to a straight line structure. The trivial case of non-overlapping time windows can be solved by visiting sites with increasing order of ready time in $O(n \log n)$ time [1]. Psaraftis conjectured that the generalized problem with both ready time and deadline is NP-hard. In this paper, we show the Psaraftis's conjecture and prove that the decision versions of the VSP on a straight line with time windows for both path and tour versions are NP-complete. In addition, we present an $O(n^2)$ algorithm based on dynamic programming approach for solving the optimization problem of a special case, in which all sites have a common ready time.

The remainder of this paper is organized as follow: section 2 gives the formulation and some notations of the problem. In section 3, we show that the decision version of the VSP on a straight line topology with time windows is NP-complete. Section 4 gives an $O(n^2)$ dynamic programming solution for solving a special case in which all sites have a common ready time for both path and tour versions. This algorithm can be generalized to tackle some variants of this special case and they are discussed in this section as well. Finally, we draw some concluding remarks in the last section.

2 Problem Formulation and Notations

VSP-WINDOW-SLP: Given a set of n sites, $S = \{s_1, s_2, \ldots, s_n\}$, in which each site s_i has a *ready time* $r(s_i)$ and *deadline* $d(s_i)$. The distance between any two sites, s_i and s_j, is denoted as $Dist(s_i, s_j)(= Dist(s_j, s_i))$, $1 \leq i, j \leq n$. A vehicle with unit speed starts at site s_1 is going to visit all sites in S.

A *schedule*, $\pi = (\pi_1, \pi_2, \ldots, \pi_n) \in \mathcal{P}_s = \{\text{set of permutations of } S\}$, is a sequence of sites so that the vehicle visits sites according to the order in π. $t_\pi(\pi_i)$ denotes the time instance at which site π_i is visited, $\forall \pi_i \in S$,

$$\begin{aligned} t_\pi(\pi_1) &= \begin{cases} \max\{r(\pi_1), Dist(s_1, \pi_1)\} & \text{if } Dist(s_1, \pi_1) \leq d(\pi_1) \\ \infty & \text{otherwise} \end{cases} \\ t_\pi(\pi_i) &= \begin{cases} \max\{r(\pi_i), t_\pi(\pi_{i-1}) + Dist(\pi_{i-1}, \pi_i)\} & \text{if } t_\pi(\pi_{i-1}) + Dist(\pi_{i-1}, \pi_i) \leq d(\pi_i) \\ \infty & \text{otherwise} \end{cases} \end{aligned}.$$

The *completion time* of a schedule π is the total time taken to visit all sites, i.e., $C_\pi = t_\pi(\pi_n)$.

A schedule π is called *feasible* if all sites can be visited in between its time window (*time window constraint*). This means that $t_\pi(\pi_i) < \infty$, $\forall \pi_i \in S$. In other words, π is feasible iff $C_\pi < \infty$ and $\pi \in \mathcal{P}_s$. Let $\mathcal{F}$ denotes the set of all feasible schedules.

An *optimal schedule* π^o is a feasible schedule with the minimal completion time, i.e., $\exists \pi^o \in \mathcal{F}$ such that

$$C_{min} = C_{\pi^o} \leq C_\pi \qquad , \forall \pi \in \mathcal{F}.$$

The *optimization problem* of the *path version* (VSP-WINDOW-SLP) is to find a π^o if $\mathcal{F} \neq \emptyset$.

Another problem restricts the vehicle return to site s_1 at the end of the trip and called a *tour version* (VSP-WINDOW-SLT). The problem formulation is exactly the same except the definition of the completion time of a schedule π is defined as

$$C_\pi^t = t_\pi(\pi_n) + Dist(s_1, \pi_n) = C_\pi + Dist(s_1, \pi_n).$$

In addition, the following notations are used in this paper,

- $i \mapsto j$ denotes site i must be visited just before j and $i \leadsto j$ if $i \mapsto s_1 \mapsto s_2 \mapsto \cdots \mapsto s_x \mapsto j$ where $s_i \in S$, or $i \mapsto j$. $i \overset{x}{\longmapsto} j$ denotes $i \mapsto j$ or $i \mapsto x \mapsto j$ where $i, j, x \in S$.

- $T(x_1 \mapsto \cdots \mapsto x_p)$ denotes the total time taken to travel along the path $x_1, x_2, \ldots, x_p$ where $x_i \in S$, i.e.,
$$T(x_1 \mapsto \cdots \mapsto x_p) = \sum_{i=1}^{p-1} Dist(x_i, x_{i+1}).$$

3 NP-hardness of VSP-WINDOW-SLP

In this section, we show the following decision question is NP-complete,

VSP-WINDOW-SLP(decision)

INSTANCE: Given a set of n sites, $\{s_1, s_2, \ldots, s_n\}$, $r(s_i)$, $d(s_i)$ and $Dist(s_j, s_{j+1})$, $1 \le i \le n, 1 \le j < n$.
QUESTION: Is there a feasible schedule for a vehicle to visit all sites and satisfy the time window constraint?

This decision problem is easier than the decision problem of the optimization problem which asks for a feasible schedule with schedule length no greater than a given constant $\mathcal{K}$. As a result, the NP-completeness of VSP-WINDOW-SLP(decision) is a stronger result. We are going to show the reduction from PARTITION which is known to be NP-complete[6].

PARTITION

INSTANCE: A finite set $I = \{1, 2, \ldots, N\}$, a "size" $a_i \ge 0$ for each $i \in I$.
QUESTION: Is there a subset $A \subseteq I$ such that

$$\sum_{i \in A} a_i = \sum_{i \in A' = I - A} a_i = \frac{1}{2} \sum_{i \in I} a_i = B?$$

3.1 A Transformation from PARTITION

Given an instance of PARTITION, we can transform it into an instance of VSP-WINDOW-SLP(decision) as follows,

Let $S = \{s, E_i, F_i, P_i, Q_i, 1 \le i \le N\}$ be the set of sites where s is the starting site, P_i and Q_i are the partition sites, E_i and F_i are the enforcer sites, and

$$n = 4N + 1, \qquad B = \frac{1}{2} \sum_{i=1}^{N} a_i, \qquad C = 2B + 1, \qquad X = NC + 2C \sum_{i=1}^{N} i + B = N^2 C + B.$$

Distance The topological structure (Fig. 1) of the VSP-WINDOW-SLP(decision) instance is

$$(Q_1, Q_2, \ldots, Q_N, s, E_1, E_2, \ldots, E_N, F_N, F_{N-1}, \ldots, F_1, P_N, P_{N-1}, \ldots, P_1) \qquad \text{and}$$

$$\begin{array}{rrcll} & Dist(Q_i, Q_{i+1}) & = & Dist(Q_N, s) = C & , 1 \le i < N \\ & Dist(s, E_1) & = & Dist(E_i, E_{i+1}) = 0 & , 1 \le i < N \\ & Dist(E_N, F_N) & = & Dist(F_i, F_{i-1}) = 0 & , 1 < i \le N \\ & Dist(F_1, P_N) & = & C + a_N/2 & \\ & Dist(P_i, P_{i-1}) & = & C + (a_i - a_{i-1})/2 & , 1 < i \le N \\ i.e., & Dist(F_1, P_i) & = & (N - i + 1)C + a_i/2 & , 1 \le i \le N \end{array}.$$

Ready Time and Deadline

$$
\begin{array}{rcl}
r(s) & = & d(s) = NC \\
r(P_i) & = & r(Q_i) = NC + 2C\sum_{N-i+2}^{N} k + (N-i+1)C \\
d(P_i) & = & d(Q_i) = X + 2C\sum_{1}^{N-i} k + (N-i+1)C + B \\
r(E_i) & = & \begin{cases} NC + 2C\sum_{1}^{N} k + B = X & \text{if } i = N \\ NC + 2C\sum_{N-i+1}^{N} k & \text{otherwise} \end{cases} \\
d(E_i) & = & NC + 2C\sum_{N-i+1}^{N} k + B \\
r(F_i) & = & \begin{cases} X + 2C\sum_{1}^{N} k + B & \text{if } i = 1 \\ X + 2C\sum_{1}^{N-i+1} k & \text{otherwise} \end{cases} \\
d(F_i) & = & X + 2C\sum_{1}^{N-i+1} k + B
\end{array}
$$

Fig. 2 shows the time windows of the above instance.

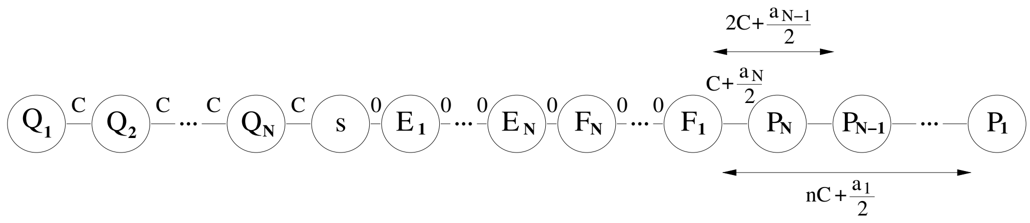

Figure 1: The topological structure of the VSP-WINDOW-SLP instance for the reduction.

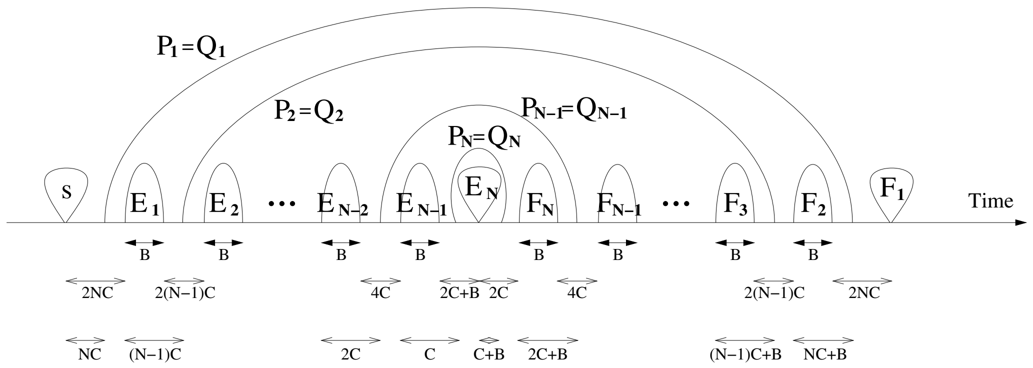

Figure 2: The ready times and deadline of the VSP-WINDOW-SLP instance for the reduction.

3.2 Intuitive Idea of the Reduction

The intuitive idea of the reduction can be explained as follows, the starting site (s) and enforcer sites (E_i and F_i, $1 \le i \le N$) have non-overlapping time windows (Fig. 2) so that there is a fixed order of visiting of these sites in any feasible schedule (proved in Lemma 1). The vehicle can only choose either P_i or Q_i in between the visiting of E_{i-1} and E_i or s and E_1 (proved in Lemma 2). It is because the partition sites Q_j and P_j ($1 \le j < i$) are too far away from the enforcer sites and the other partition sites, Q_k and P_k ($i < k \le N$) have not been ready yet (Fig. 3).

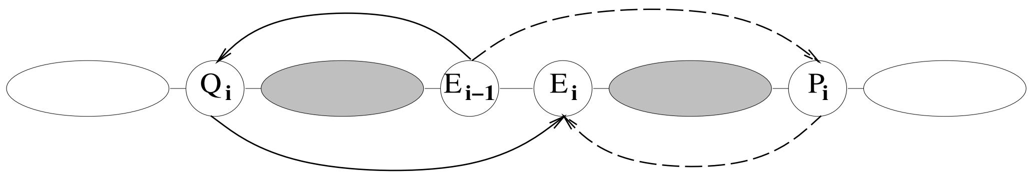

Figure 3: The solid and dotted lines represent the two possible routes. The partition sites in the blank region are too far away from E_{i-1} and E_i. The partition sites in the shaded region have not been ready before the deadline of E_i.

Similarly, the vehicle can only choose either P_i or Q_i in between the visiting of F_{i+1} and F_i or E_N and F_N (proved in Lemma 3). The vehicle is now constrainted by the deadline of the partition sites (Fig. 4).

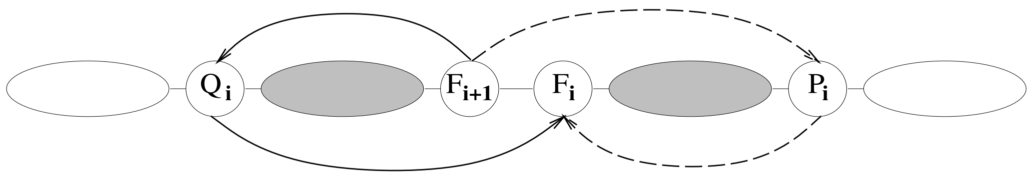

Figure 4: The solid and dotted lines represent the two possible routes. The partition sites in the blank region are too far away from F_{i+1} and F_i. The partition sites in the shaded region should have been visited due to the deadline constraint.

Lastly, sites s, E_n and F_1 should be visited at their deadline, i.e., $t = NC$, X and $2X - NC$ respectively. The elapsed times between the visiting times of these sites are

$$X - NC = 2C \sum_{1}^{N} i + B.$$

As a result, the set of partition sites, $\{P_i : 1 \leq i \leq N\}$, is partitioned into two parts, P_L and P_R (Fig. 5), such that

$$\sum_{P_i \in P_L} a_i = \sum_{P_i \in P_R} a_i = B.$$

These two sets, P_L and P_R, correspond to the solution of the PARTITION instance.

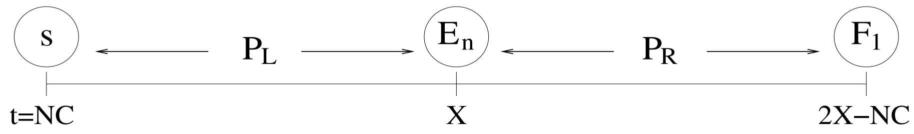

Figure 5: Relationship between the PARTITION instance and the constructed VSP-WINDOW-SLP instance.

3.3 NP-completeness Proof

Lemma 1 *s must be the first site to be visited and*

$$s \rightsquigarrow E_1 \rightsquigarrow E_2 \rightsquigarrow \cdots \rightsquigarrow E_N \rightsquigarrow F_N \rightsquigarrow F_{N-1} \rightsquigarrow \cdots \rightsquigarrow F_1$$

in any feasible schedule.

Proof. It is easily observed that the time windows of the starting site, s, and the enforcer sites (E_i and F_i, $1 \leq i \leq N$) are non-overlapping (Fig. 2). Therefore, there are a strict sequence of visiting of these sites in any feasible schedule. □

Lemma 2 *The sequence of sites of any feasible schedule before the visiting of E_N is*

$$s \xmapsto{R_1} E_1 \xmapsto{R_2} E_2 \xmapsto{R_3} \cdots \xmapsto{R_N} E_N,$$

where $R_i \in \{P_i, Q_i\}$.

Proof. Firstly, we are going to prove P_j or Q_j cannot be visited before E_i in any feasible schedule, $\forall i, j, 1 \leq i < j \leq N$.

It is obvious that $r(P_i) < r(P_{i+1}), 1 \leq i < N$. Thus, $\forall i, j, 1 \leq i < j \leq N$,

$$\begin{aligned} r(P_j) - d(E_i) &\geq r(P_{i+1}) - d(E_i) \\ &= NC + 2C \sum_{N-i+1}^{N} k + (N-i)C - (NC + 2C \sum_{N-i+1}^{N} k + B) \\ &= (N-i)C - B > C - B > 0 \qquad\qquad (C > B). \end{aligned}$$

Therefore, $E_i \rightsquigarrow P_j$, $1 \leq i < j \leq N$.

Similar result holds for Q_i because $r(P_i) = r(Q_i)$.

Secondly, we are going to prove P_j or Q_j cannot be visited before E_i in any feasible schedule, $\forall i, j, 1 \leq j < i \leq N$. For $2 \leq i \leq N$,

$$\begin{aligned} d(E_i) - r(E_{i-1}) &= NC + 2C \sum_{N-i+1}^{N} k + B - (NC + 2C \sum_{N-i+2}^{N} k) \\ &= 2C(N-i+1) + B. \end{aligned} \tag{1}$$

For $1 \leq j < i \leq N$, $1 \leq k \leq N$, $Dist(E_i, P_j) \geq Dist(E_i, Q_j)$, $Dist(E_k, Q_j) \geq Dist(E_k, Q_{i-1})$,

$$\begin{aligned} T(E_{i-1} \mapsto P_j \mapsto E_i) &\geq T(E_{i-1} \mapsto Q_j \mapsto E_i) \\ &\geq T(E_{i-1} \mapsto Q_{i-1} \mapsto E_i) \\ &= 2C(N-i+2) \\ &> 2C(N-i+1) + B \\ &= d(E_i) - r(E_{i-1}) \qquad\qquad (\text{by (1)}). \end{aligned}$$

Therefore, we cannot visit P_j or Q_j before visiting E_i, $1 \leq j < i \leq N$.

By combining the above two result, the only possible choice between the visiting of E_{i-1} and E_i are P_i or Q_i, $1 < i \leq N$. Now we are going to prove that we cannot visit P_i followed by Q_i (or vice versa),

$$\begin{aligned} d(E_1) - r(s) &= NC + 2NC + B - NC \\ &= 2NC + B. \end{aligned} \tag{2}$$

$$
\begin{aligned}
T(s \mapsto P_1 \mapsto Q_1 \mapsto E_1) &= T(s \mapsto Q_1 \mapsto P_1 \mapsto E_1) \\
&= 4NC + a_1 \\
&> 2NC + B \\
&= d(E_1) - r(s) \qquad \text{(by (2))}.
\end{aligned}
$$

For $2 \leq i \leq N$,

$$
\begin{aligned}
T(E_{i-1} \mapsto P_i \mapsto Q_i \mapsto E_i) &= T(E_{i-1} \mapsto Q_i \mapsto P_i \mapsto E_i) \\
&= 4C(N-i+1) + a_i \\
&> 2C(N-i+1) + B \\
&= d(E_i) - r(E_{i-1}) \qquad \text{(by (1))}.
\end{aligned}
$$

We can visit either P_i or Q_i (but not both) in between the visiting of E_{i-1} and E_i. By Lemma 1, the sequence of sites in any feasible schedule before visiting E_N is

$$
s \xmapsto{R_1} E_1 \xmapsto{R_2} E_2 \xmapsto{R_3} \cdots \xmapsto{R_N} E_N,
$$

where $R_i \in \{P_i, Q_i\}$. □

Lemma 3 *The sequence of sites in any feasible schedule after the visiting of E_N is*

$$
E_N \xmapsto{R_N} F_N \xmapsto{R_{N-1}} F_{N-1} \xmapsto{R_{N-2}} \cdots \xmapsto{R_1} F_1,
$$

where $R_i \in \{P_i, Q_i\}$.

Proof. Firstly, we are going to prove P_j and Q_j must be visited before F_i in any feasible schedule, $\forall i, j, 1 \leq i \leq j \leq N$.

It is obvious that $d(P_i) \geq d(P_j), 1 \leq i \leq j \leq N$. Thus, $\forall i, j, 1 \leq i \leq j \leq N$,

$$
\begin{aligned}
r(F_i) - d(P_j) &\geq r(F_i) - d(P_i) \\
&\geq X + 2C \sum_{1}^{N-i+1} k - (X + 2C \sum_{1}^{N-i} k + (N-i+1)C + B) \\
&= 2C(N-i+1) - (N-i+1)C - B \\
&= (N-i+1)C - B > 0.
\end{aligned}
$$

Therefore, P_j (and thus Q_j) must be visited before F_i, $\forall i, j$ and $1 \leq i \leq j \leq N$.

Secondly, we are going to prove P_j or Q_j cannot be visited before F_i in any feasible schedule, $\forall i, j, 1 \leq j < i \leq N$. For $1 \leq i < N$,

$$
\begin{aligned}
d(F_i) - r(F_{i+1}) &= X + 2C \sum_{K=1}^{N-i+1} k + B - (X + 2C \sum_{1}^{N-i} k) \\
&= 2C(N-i+1) + B
\end{aligned} \tag{3}
$$

For $1 \leq j < i \leq N$, $Dist(F_i, P_j) \geq Dist(F_i, Q_j)$ and $Dist(F_i, Q_j) \geq Dist(F_i, Q_{i-1})$,

$$
\begin{aligned}
T(F_{i+1} \mapsto P_j \mapsto F_i) &\geq T(F_{i+1} \mapsto Q_j \mapsto F_i) \\
&\geq T(F_{i+1} \mapsto Q_{i-1} \mapsto F_i) \\
&= 2C(N-i+2) \\
&> 2C(N-i+1) + B \\
&= d(F_i) - r(F_{i+1}) \qquad \text{(by (3))}.
\end{aligned}
$$

Therefore, we cannot visit P_j or Q_j before visiting F_i, $1 \le j < i < N$.

For F_N,

$$\begin{aligned} d(F_N) - r(E_N) &= X + 2C - X \\ &= 2C. \end{aligned} \tag{4}$$

$$\begin{aligned} T(E_N \mapsto P_j \mapsto F_N) &\ge T(E_N \mapsto Q_j \mapsto F_N) \\ &\ge T(E_N \mapsto Q_{N-1} \mapsto F_N) \\ &= 4C \\ &> d(F_N) - r(E_N) \qquad \text{(by (4))}. \end{aligned}$$

By the above two result, we cannot visit P_i or Q_i before F_{i+1} but they must be visited before the visiting of F_i. We are now going to prove that we cannot visit both P_i and Q_i in between the visiting of F_{i+1} and F_i,

$$\begin{aligned} T(E_N \mapsto P_N \mapsto Q_N \mapsto F_N) &= T(E_N \mapsto Q_N \mapsto P_N \mapsto F_N) \\ &= 4C + a_N \\ &> d(F_N) - r(E_N) \qquad \text{(by (4))}. \end{aligned}$$

For $1 \le i < N$,

$$\begin{aligned} T(F_{i+1} \mapsto P_i \mapsto Q_i \mapsto F_i) &= T(F_{i+1} \mapsto Q_i \mapsto P_i \mapsto F_i) \\ &= 4C(N - i + 1) + a_i \\ &\ge 2C(N - i + 1) + 4C + a_i \\ &> 2C(N - i + 1) + B \\ &= d(F_i) - r(F_{i+1}) \qquad \text{(by (3))}. \end{aligned}$$

We can visit either P_i or Q_i (but not both) in between the visiting of F_{i+1} and F_i, By Lemma 1, the sequence of sites in any feasible schedule after visiting E_N is

$$E_N \xmapsto{R_N} F_N \xmapsto{R_{N-1}} F_{N-1} \xmapsto{R_{N-2}} \cdots \xmapsto{R_1} F_1,$$

where $R_i \in \{P_i, Q_i\}$. □

Lemma 4 *If π is a feasible schedule, then*

$$\pi = (s, R_1, E_1, R_2, E_2, \ldots, R_N, E_N, R'_N, F_N, R'_{N-1}, F_{N-1}, \ldots, R'_1, F_1),$$

where

$$\begin{aligned} R_i &= P_i \text{ or } Q_i, \\ R'_i &= \begin{cases} P_i & \text{if } R_i = Q_i \\ Q_i & \text{if } R_i = P_i \end{cases}. \end{aligned}$$

Proof. By Lemma 1, Lemma 2, Lemma 3 and all sites must be visited □

Lemma 5 *PARTITION has a solution iff the corresponding VSP-WINDOW-SLP(decision) instance has a feasible schedule.*

Proof. For any schedule π

$$\pi = (s, R_1, E_1, R_2, E_2, \ldots, R_N, E_N, R'_N, F_N, R'_{N-1}, F_{N-1}, \ldots, R'_1, F_1),$$

where

$$\begin{aligned} R_i &= P_i \text{ or } Q_i, \\ R'_i &= \begin{cases} P_i & \text{if } R_i = Q_i \\ Q_i & \text{if } R_i = P_i \end{cases}. \end{aligned}$$

Let
$$\begin{aligned} A_\pi &= \{i : R_i = P_i \text{ in } \pi, 1 \le i \le N\} \\ A'_\pi &= \{i : R'_i = P_i \text{ in } \pi, 1 \le i \le N\} \text{ and} \end{aligned}$$

$$A^i_\pi = \sum_{j \le i, j \in A_\pi} a_j \quad \text{and} \quad A'^i_\pi = \sum_{j \le i, j \in A'_\pi} a_j.$$

Then,

$$\begin{aligned} t_\pi(s) &= NC \\ t_\pi(R_1) &= \max\{t_\pi(s) + Dist(s, R_1), r(R_1)\} \\ t_\pi(R_i) &= \max\{t_\pi(E_{i-1}) + Dist(E_{i-1}, R_i), r(R_i)\} && ,2 \le i \le N \\ t_\pi(E_i) &= \max\{t_\pi(R_i) + Dist(E_i, R_i), r(E_i)\} && ,1 \le i \le N \\ t_\pi(R'_N) &= \max\{t_\pi(E_N) + Dist(E_N, R'_N), r(R'_N)\} \\ t_\pi(R'_i) &= \max\{t_\pi(F_{i+1}) + Dist(F_{i+1}, R'_i), r(R'_i)\} && ,1 \le i < N \\ t_\pi(F_i) &= \max\{t_\pi(R'_i) + Dist(F_i, R'_i), r(R'_i)\} && ,1 \le i \le N \end{aligned} .$$

If $A^N_\pi \le B$, then

$$\begin{aligned} t_\pi(s) &= NC \\ t_\pi(R_i) &= \begin{cases} NC + 2C\sum_{N-i+2}^{N} k + (N-i+1)C + A^{i-1}_\pi + \frac{a_i}{2} & \text{if } R_i = P_i \\ NC + 2C\sum_{N-i+2}^{N} k + (N-i+1)C + A^{i-1}_\pi & \text{otherwise} \end{cases} \\ t_\pi(E_i) &= NC + 2C\sum_{N-i+1}^{N} k + A^i_\pi \\ t_\pi(R'_i) &= \begin{cases} X + 2C\sum_{1}^{N-i+1} k + (N-i+1)C + A'^{i-1}_\pi + \frac{a_i}{2} & \text{if } R'_i = P_i \\ X + 2C\sum_{N-i+2}^{N} k + (N-i+1)C + A'^{i-1}_\pi & \text{otherwise} \end{cases} \\ t_\pi(F_i) &= X + 2C\sum_{1}^{N-i+1} k + A'^i_\pi. \end{aligned}$$

"$\Rightarrow$" Let A and $A'(= I - A)$ be a solution of the PARTITION problem, construct a schedule π such that,

$$\pi = (s, R_1, E_1, R_2, E_2, \ldots, R_N, E_N, R'_N, F_N, R'_{N-1}, F_{N-1}, \ldots, R'_1, F_1),$$

where

$$\begin{aligned} R_i &= \begin{cases} P_i & \text{if } i \in A \\ Q_i & \text{if } i \in A' \end{cases} \\ R'_i &= \begin{cases} P_i & \text{if } R_i = Q_i \\ Q_i & \text{if } R_i = P_i \end{cases}. \end{aligned}$$

Then, $A_\pi = A$, $A'_\pi = A' = I - A$ and $A^N_\pi = A'^N_\pi = B$,

$$t_\pi(s) = NC \le d(s).$$

If $R_i = P_i$,

$$\begin{aligned} t_\pi(R_i) &= NC + 2C \sum_{N-i+2}^{N} k + (N-i+1)C + A_\pi^{i-1} + \frac{a_i}{2} \\ &\le NC + 2C \sum_{N-i+2}^{N} k + (N-i+1)C + A_\pi^{N} \qquad (i \in A_\pi) \\ &= d(R_i). \end{aligned}$$

If $R_i = Q_i$,

$$\begin{aligned} t_\pi(R_i) &= NC + 2C \sum_{N-i+2}^{N} k + (N-i+1)C + A_\pi^{i-1} \\ &\le NC + 2C \sum_{N-i+2}^{N} k + (N-i+1)C + A_\pi^{N} \\ &= d(R_i) \end{aligned}$$

and $t_\pi(R_i) \ge r(R_i)$, i.e., all R_i can be visited properly.

$$\begin{aligned} t_\pi(E_i) &= NC + 2C \sum_{N-i+1}^{N} k + A_\pi^{i} \\ &\le NC + 2C \sum_{N-i+1}^{N} k + A_\pi^{N} \\ &= d(E_i). \end{aligned}$$

For $1 \le i < N$,

$$\begin{aligned} t_\pi(E_i) &= NC + 2C \sum_{N-i+1}^{N} k + A_\pi^{i} \\ &\ge NC + 2C \sum_{N-i+1}^{N} k \\ &= r(E_i). \end{aligned}$$

For E_N,

$$\begin{aligned} t_\pi(E_N) &= NC + 2C \sum_{1}^{N} k + A_\pi^{N} \\ &= NC + 2C \sum_{1}^{N} k + B \\ &= r(E_N). \end{aligned}$$

Therefore, all E_i can be visited properly.

If $R_i' = P_i$,

$$\begin{aligned} t_\pi(R_i') &= X + 2C \sum_{1}^{N-i+1} k + (N-i+1)C + A_\pi'^{i-1} + \frac{a_i}{2} \\ &\le X + 2C \sum_{1}^{N-i+1} k + (N-i+1)C + A_\pi'^{N} \qquad (i \in A_\pi') \\ &= d(R_i'). \end{aligned}$$

If $R'_i = Q_i$,

$$\begin{aligned} t_\pi(R_i) &= X + 2C\sum_1^{N-i+1} k + (N-i+1)C + A'^{i-1}_\pi \\ &\leq X + 2C\sum_1^{N-i+1} k + (N-i+1)C + A'^{N}_\pi \\ &= d(R'_i) \end{aligned}$$

and $t_\pi(R'_i) \geq r(R'_i)$, i.e., all R'_i can be visited properly.

$$\begin{aligned} t_\pi(F_i) &= X + 2C\sum_1^{N-i+1} k + A'^{i}_\pi \\ &\leq X + 2C\sum_1^{N-i+1} k + A'^{N}_\pi \\ &= d(F_i). \end{aligned}$$

For $1 < i \leq N$,

$$\begin{aligned} t_\pi(F_i) &= X + 2C\sum_1^{N-i+1} k + A'^{i}_\pi \\ &\geq X + 2C\sum_1^{N-i+1} k \\ &= r(F_i). \end{aligned}$$

For F_1,

$$\begin{aligned} t_\pi(F_1) &= X + 2C\sum_1^{N} k + A'^{N}_\pi \\ &= X + 2C\sum_1^{N} k + B \\ &= r(F_1). \end{aligned}$$

Therefore, all F_i can be visited properly. As a result, π is a feasible schedule.

"$\Leftarrow$" Suppose on the contrary, the PARTITION problem has no solution and there is a feasible schedule for the VSP-WINDOW-SLP(decision) problem, i.e., $\forall A \subseteq I$ and $A' = I - A$,

$$\sum_{i\in A} a_i \neq \sum_{i\in A'} a_i \neq B. \tag{5}$$

Since π is a feasible schedule, $t_\pi(E_N) \leq d(E_N)$, i.e.,

$$\begin{aligned} &t_\pi(E_N) = NC + 2C\textstyle\sum_1^N k + A^N_\pi \leq NC + 2C\sum_1^N k + B \\ \Rightarrow\ &A^N_\pi \leq B \end{aligned}.$$

By (5),

$$\begin{aligned} &A^N_\pi = \textstyle\sum_{i\in A_\pi} a_i \neq B \\ \Rightarrow\ &A^N_\pi < B \\ \Rightarrow\ &A'^N_\pi = 2B - A^N_\pi > B \end{aligned},$$

$$
\begin{aligned}
t_\pi(F_1) &= X + 2C\sum_1^N k + A'^N_\pi \\
&> X + 2C\sum_1^N k + B \\
&= d(F_1),
\end{aligned}
$$

which is contradictory to the fact that π is a feasible schedule. $\square$

Theorem 6 *VSP-WINDOW-SLP(decision) is NP-complete.*

Proof. It is obvious that VSP-WINDOW-SLP(decision) is NP and the transformation in section 3.1 can be done in polynomial time. By Lemma 5, VSP-WINDOW-SLP(decision) is NP-complete. $\square$

Theorem 7 *The decision problem of asking for a feasible schedule for the tour version (VSP-WINDOW-SLT) is NP-complete.*

Proof. It is obvious that π is a feasible schedule for the path version iff it is a feasible schedule for the tour version. Thus, the reduction is applicable to the tour version, VSP-WINDOW-SLT, so that the decision problem of the tour version is NP-complete by Theorem 6. $\square$

4 Polynomial Time Algorithm for the VSP-WINDOW on a straight line with Common Ready Time

The result in the last section implies that there is no hope for finding an efficient algorithm to solve the general VSP-WINDOW-SLP problem unless $P = NP$. In this section, we try to develop an efficient algorithm that can solve a special case of VSP-WINDOW-SLP in which all sites have the same ready time, i.e., $r(i) = r(j), \forall i, j \in S$. For the rest of this section, the sites are labelled as $1, 2, \ldots, n$ so that sites i and $i+1$ are connected adjacently, $\forall i, 1 \leq i < n$ and the vehicle is initially located at site 1. If $r(i) = 0, \forall i, 1 \leq i \leq n$, a trivial solution for an optimal schedule is $\pi = (1, 2, \ldots, n)$ for both path and tour versions provided that the problem has a feasible solution. The time complexity is $O(n)$. For the general case, the feasible schedule is characterized as follows,

Lemma 8 *In the path version, VSP-WINDOW-SLP, for all feasible schedules, π, there exists a feasible schedule π' s.t. $\forall 1 < i \leq n$,*

$$\pi'_i = \min_{1 \leq j < i}\{\pi'_j\} - 1 \; or \; \max_{1 \leq j < i}\{\pi'_j\} + 1,$$

i.e., at any instant, the visited sites form a consecutive region on the straight line and the last visited site is either the leftmost or rightmost site in this region. In addition,

$$C_{\pi'} + Dist(\pi'_n, \pi_n) \leq C_\pi,$$

i.e., the completion time of π' plus the time taken for the vehicle returns to the last visited site in π is no greater than the completion time of π.

Proof. If $\mathcal{F} \neq \emptyset$, there exists an feasible schedule $\pi \in \mathcal{F}$. Assume π does not satisfy the criterion, we are going to construct another schedule, π' s.t. $\pi'_i = \min_{1 \leq j < i}\{\pi'_j\} - 1$ or $\max_{1 \leq j < i}\{\pi'_j\} + 1, \forall i, 1 < i \leq n$. Let j be the smallest index in π such that $1 < j \leq n$,

$$\pi_j \neq \min_{1 \leq k < j}\{\pi_k\} - 1 \text{ and } \pi_j \neq \max_{1 \leq k < j}\{\pi_k\} + 1.$$

Equivalently,

$$\pi_j > \max_{1 \leq k < j}\{\pi_k\} + 1 \text{ and } \pi_j < \min_{1 \leq k < j}\{\pi_k\} - 1.$$

Case 1: $\pi_j > \max_{1 \leq k < j}\{\pi'_k\} + 1$.

Since π_{j-1} has been visited and all sites have the same ready time, we can visit any site with no waiting time and thus

$$t_\pi(\pi_j) = t_\pi(\pi_{j-1}) + Dist(\pi_{j-1}, \pi_j).$$

Let $\pi_x = \max_{1 \leq k < j}\{\pi_k\} + 1$ in π, then we can construct a schedule π" s.t.

$$\pi" = (\pi_1, \ldots, \pi_{j-1}, \pi_x, \pi_j, \ldots, \pi_{x-1}, \pi_{x+1}, \ldots, \pi_n),$$

$$\begin{array}{rcll} t_{\pi"}(\pi_x) & = & t_\pi(\pi_{j-1}) + Dist(\pi_{j-1}, \pi_x) & \\ & \leq & t_\pi(\pi_{j-1}) + Dist(\pi_{j-1}, \pi_j) + Dist(\pi_j, \pi_x) & (\pi_j > \pi_x) \\ & \leq & t_\pi(\pi_x) \leq d(\pi_x) & (\pi \in \mathcal{F}) \\ t_{\pi"}(\pi_j) & = & t_\pi(\pi_{j-1}) + Dist(\pi_{j-1}, \pi_x) + Dist(\pi_x, \pi_j) & (\pi_x < \pi_j) \\ & = & t_\pi(\pi_j) \leq d(\pi_j) & (\pi \in \mathcal{F}) \end{array}.$$

$\forall y, j \leq y \leq x - 1$,

$$t_{\pi"}(\pi_y) = t_\pi(\pi_y). \tag{6}$$

$\forall y, x + 1 \leq y \leq n$,

$$t_{\pi"}(\pi_y) \leq t_\pi(\pi_y). \tag{7}$$

Thus, $\pi" \in \mathcal{F}$.

If $x = n$, π_x is the last visited site in π, then $\pi"_n = \pi_{x-1}$ and

$$\begin{array}{rcll} C_{\pi"} + Dist(\pi"_n, \pi_n) & = & t_{\pi"}(\pi_{x-1}) + Dist(\pi_{x-1}, \pi_x) & \\ & = & t_\pi(\pi_{x-1}) + Dist(\pi_{x-1}, \pi_x) & \text{(by (6))} \\ & = & t_\pi(\pi_x) = C_\pi & \end{array}.$$

If $x \neq n$, $\pi"_n = \pi_n$, then

$$\begin{array}{rcll} C_{\pi"} + Dist(\pi"_n, \pi_n) & = & t_{\pi"}(\pi_n) + Dist(\pi_n, \pi_n) & \\ & \leq & t_\pi(\pi_n) & \text{(by (7))} \\ & = & C_\pi & \end{array}.$$

As a result, $C_{\pi"} + Dist(\pi"_n, \pi_n) \leq C_\pi$.

Case 2: $\pi_j < \min_{1 \leq k < j}\{\pi_k\} - 1$. Similar to Case 1 and $\pi"_j = \min_{1 \leq k < j}\{\pi_k\} - 1$.

In both case, $\pi"_j = \min_{1 \leq j < i}\{\pi'_j\} - 1$ or $\max_{1 \leq j < i}\{\pi'_j\} + 1$ and $C_{\pi"} + Dist(\pi"_n, \pi_n) \leq C_\pi$. Repeat the above procedure, we can eventually construct a feasible schedule π' s.t. $\forall i, 1 < i \leq n$,

$$\pi'_i = \min_{1 \leq j < i}\{\pi'_j\} - 1 \text{ or } \max_{1 \leq j < i}\{\pi'_j\} + 1 \qquad \text{and} \qquad C_{\pi'} + Dist(\pi'_n, \pi_n) \leq C_\pi.$$

□

By Lemma 8, we can solve the VSP-WINDOW-SLP problem with common ready time by an algorithm based on dynamic programming as follows,

Algorithm A (Dynamic Programming Approach)

Input: *A set of n sites, $\{1, 2, \ldots, n\}$, their ready time, $r(i)$, deadline, $d(i)$ and the distance between adjacent sites, $Dist(j, j+1)$, $1 \leq i \leq n$ and $1 \leq j < n$.*

Output: *The minimal completion time if $\mathcal{F} \neq \emptyset$, otherwise ∞.*

(1) for $i = 1$ to n //boundary conditions
(2) $\quad V(i, i, i) = \delta_i(\max\{r(i), Dist(1, i)\})$
(3) $\quad V(n+1, n+1, i) = V(i, n+1, i) = \infty$
(4) for $l = 2$ to n //l is the number of visited sites
(5) $\quad$ for $i = 1$ to $n - l + 1$ //i is the visited site with the smallest index
(6) $\quad\quad j = i + l - 1$ //j is the visited site with the largest index
(7) $\quad\quad V(i, i, j) = \min\{\delta_i(Dist(i+1, i) + V(i+1, i+1, j)),$
$\quad\quad\quad\quad \delta_i(Dist(j, i) + V(j, i+1, j))\}$
(8) $\quad\quad V(j, i, j) = \min\{\delta_j(Dist(j-1, j) + V(j-1, i, j-1)),$
$\quad\quad\quad\quad \delta_j(Dist(i, j) + V(i, i, j-1))\}$
(9) $C_{min} = \min\{V(1, 1, n), V(n, 1, n)\}$
(10) return C_{min}

In Algorithm A, $V(k, i, j)$ denotes the time instance that sites $i, i+1, \ldots, j$ are visited and the last visited site is k, $i \leq k \leq j$. The function $\delta_i(x)$ is defined as follows,

$$\delta_i(x) = \begin{cases} x & \text{if } x \leq d(i) \\ \infty & \text{otherwise} \end{cases}.$$

There are altogether $O(n^2)$ iterations. $V(i, i, j)$ and $V(j, i, j)$ can be calculated in constant time by using an $O(n^2)$ array (line 7 and 8). The time and space complexities of the above algorithm are $O(n^2)$. In addition, we can record the optimal schedule by some simple modifications.

Theorem 9 *Algorithm A returns the minimal completion time for the VSP-WINDOW-SLP with common ready time in $O(n^2)$ time.*

Proof. By Lemma 8, for all feasible schedule π, there exists a feasible schedule π' such that the next site to be visited is the first site which is located either left or right of all visited sites and $C_{\pi'} + Dist(\pi'_n, \pi) \leq C_\pi$. As are result, if $\mathcal{F} \neq \emptyset$, there exists an optimal schedule π^o such that it satisfies the above criterion because $\forall \pi \in \mathcal{F}$,

$$C_{\pi^o} \leq C_{\pi^o} + Dist(\pi^o_n, \pi_n) \leq C_\pi.$$

Thus, we can only consider the cases that the last visited site has either the smallest or largest index ($V(i, i, j)$ and $V(j, i, j)$ in line 7 and 8). It is obvious that C_{min} in Algorithm A is the optimal completion time. □

Theorem 10 *We can find the optimal completion time for VSP-WINDOW-SLT with common ready time in $O(n^2)$ time.*

Proof. By Lemma 8, for all $\pi \in \mathcal{F}$, there exists a feasible schedule, π', such that the next site to be visited is the first site which is located either left or right of all visited sites and $C_{\pi'} + Dist(\pi'_n, \pi) \leq C_\pi$. In the tour version, VSP-WINDOW-SLT, the completion time of a schedule π is $C^t_\pi = C_\pi + Dist(\pi_n, 1)$. Thus

$$\begin{aligned} C^t_{\pi'} &= C_{\pi'} + Dist(\pi'_n, 1) \\ &\leq C_{\pi'} + Dist(\pi'_n, \pi_n) + Dist(\pi_n, 1) \\ &\leq C_\pi + Dist(\pi_n, 1) \qquad \text{(by Lemma 8)} \\ &= C^t_\pi \end{aligned},$$

i.e., there exists an optimal schedule such that it satisfies the criterion in Lemma 8. As a result, Algorithm A is applicable to the tour version by modifying line 9 and 10 as follows,

$$\begin{array}{ll} \text{line 9:} & C^t_{min} = \min\{V(1,1,n), V(n,1,n) + Dist(1,n)\} \\ \text{line 10:} & \text{return } C^t_{min} \end{array}.$$

□

In addition, this algorithm can be extended to the case in which the vehicle starts at time $t = 0$ not at site 1, but at some interior site s between 1 and n. The boundary conditions (line 2 and 3) are changed as follows, $\forall i, 1 \leq i \leq n$,

$$V(i,i,i) = \delta_i(\max\{r(i), Dist(s,i)\}).$$

For the tour version, the additional modification is line 9,

$$C^t_{min} = \min\{V(1,1,n), V(n,1,n) + Dist(s,n)\}.$$

Algorithm A is only applicable to the cases that all sites have a common ready time, since Lemma 8 will not be valid if sites have different ready times. The reason is that the original schedule, π, in Lemma 8 cannot be replaced by the new route, π', or else the vehicle may have to wait at site π_x (due to the different ready time) and thus the deadline of π_j may be missed.

5 Conclusion

In this paper, we study the computational complexities of the single-vehicle scheduling problems (VSP) on a straight line with time window constraints. In section 3, we prove the conjecture given by Psaraftis[1] and show that the decision problems of the vehicle scheduling problem (VSP) on a straight line with time window constraints for both path (VSP-WINDOW-SLP) and tour (VSP-WINDOW-SLT) versions are NP-complete. An extension to the case in which the vehicle is required to spend a known "service" time (called *handling time*) at each site is also an interesting problem. It is easily observed that the SEQUENCING WITHIN INTERVALS[6], which is a strongly NP-complete problem, is a restricted case of the VSP on a straight line with time windows and handling time. Thus the extension of VSP-WINDOW-SLP and VSP-WINDOW-SLT with handling time are NP-hard in the strong sense. If the problem has only ready time, deadline or handling time, it can be solved in polynomial time. The path and tour versions of the VSP on a straight line with ready time only can be solved in $O(n^2)$ and $O(n)$ times respectively, where n is the number of sites[1]. The trivial cases of the VSP on a straight line with deadline or handling time only can be solved in $O(n)$ time for both path and tour versions. The optimal schedule is a trip visiting all sites successively from the starting site to the other end of the straight line topology. However, the status of the VSP on a straight line with ready and handling times only (or deadline and handling time) is still open. Psaraftis conjectured they are NP-hard[1]. Besides, it remains to examine whether the VSP-WINDOW-SLP and VSP-WINDOW-SLT are strongly NP-hard or not. The status of different single-vehicle scheduling problems on a straight line topological structure is summarized in Table 1.

	Path version	Tour version
ready time (r)	$O(n^2)^\dagger$	$O(n)^\dagger$
deadline (d)	$O(n)$	$O(n)$
handling time (h)	$O(n)$	$O(n)$
r and d	NP-hard	NP-hard
r and h	$?^\ddagger$	?
d and h	?	?
r,d and h	NP-hard in the strong sense	NP-hard in the strong sense

†: By Psaraftis's result[1].

‡: A ? denotes an unknown status.

Table 1: The computation complexities of the different variants of the VSP on a straight line topology.

A special case of the VSP on a straight line with time windows, in which all sites have a common ready time, is investigated in section 4. We present a dynamic programming algorithm, Algorithm A, for solving both the path and tour versions. The time and space complexities of Algorithm A are $O(n^2)$. Furthermore, our algorithm can be extended to tackle problems in which the vehicle is initially located at some given interior site on the straight line.

References

[1] H. Psaraftis, M. Solomon, L. Magnanti, and Tai-Up Kim. "Routing and Scheduling on a Shoreline with Release Times". *Management Science*, vol 36:pp. 212–223, February 1990.

[2] L. Bodin, B. Golden, A. Assad, and M. Ball. "Routing and Scheduling of Vehicles and Crews: The State of the Art". *Computers and Operations Research*, vol 10:pp. 62–212, 1983.

[3] H. Psaraftis. "An Exact Algorithm for the Single Vehicle Dial-A-Ride Problem with Time Windows". *Transportation Science*, vol 17:pp. 351–357, 1983.

[4] Marius M. Solomon. "Algorithms for the vehicle routing and Scheduling problems with Time Window Constraints". *Operations Research*, vol 35:pp. 254–265, 1987.

[5] Y. Karuno, H. Nagamochi, and T. Ibaraki. "Vehicle Scheduling on a Tree with Release and Handling Times". In *"Lecture Notes in Computer Science(762)"*, pages 486–495, 1993.

[6] M.R. Garey and D.S. Johnson. *"Computers and Intractability: A Guide to the Theory of NP-Completeness"*. Freeman, 1979.