Speed optimization over a path with heterogeneous arc costs

Qie He[*], Xiaochen Zhang[†], Kameng Nip[‡]

March 16, 2017

Abstract

The speed optimization problem over a path aims to find a set of speeds over each arc of the given path to minimize the total cost, while respecting the time-window constraint at each node and speed limits over each arc. In maritime transportation, the cost represents fuel cost or emissions, so study of this problem has significant economic and environmental impacts. To accommodate different fuel and emission models, we allow the dependence of the cost on the speed to be a general continuously differentiable and strictly convex function, and different across the arcs. We develop an efficient algorithm that is able to solve instances of 1,000 nodes in less than a second. The algorithm is 20 to 100 times faster than a general convex optimization solver on test instances and requires much less memory. The solutions found at intermediate steps of our algorithm also provide some insights to ship planners on how to balance the operating cost and service quality.

Keywords. maritime transportation; bunker consumption; emission control; speed optimization

1 Introduction

Maritime transportation plays an important role in international trade. The volume of seaborne shipments surpasses 10 billion in 2015, accounting for more than four fifths of total world merchandise trade [37]. In maritime transportation, vessel sailing speed has a significant impact on bunker consumption, emissions, and transit time, and therefore is a critical operational decision [32]. It has been shown that reducing sailing speed can lead to substantial savings in bunker consumption [42, 32, 33], which constitutes a significant portion of the operating cost for a shipping company [28]. Meanwhile, the greenhouse gas emissions are positively correlated to the bunker consumption, so reducing sailing speed also effectively reduce emissions and the negative impact on the environment [12]. On the other hand, lower sailing speed means longer transit and delivery time, which affects the service quality and other operational decisions [27, 41, 26, 32, 44]. Therefore, speed optimization is a basic problem in maritime transportation to understand the trade-off among operating cost, environmental impact, and service quality.

In this paper, we study a low-level operational problem regarding the vessel speed decisions. To be more specific, we would like to answer the following research question:

Speed Optimization Over a Path (SOOP): Given a sequence of nodes over a path, what is the optimal speed between each pair of consecutive nodes such that the total cost over the path is minimized and each customer is served within a prescribed time window?

[*]Department of Industrial and Systems Engineering, University of Minnesota. Email: qhe@umn.edu.
[†]Department of Industrial and Systems Engineering, University of Minnesota. Email: zhan4487@umn.edu.
[‡]Department of Mathematical Sciences, Tsinghua University. Email: njm13@mails.tsinghua.edu.cn.

This question was initially studied by [13] in the context of tramp shipping. In [13], a tramp trip is scheduled to visit a sequence of ports over a route, and the goal is to find the sailing speed over each leg of the route to minimize the fuel emissions. The fuel consumption rate is assumed to be a convex function of the sailing speed, and remains the same over all legs of the route.

In this paper, we assume the cost per unit distance traveled to be *different across the arcs*. We consider heterogeneous arc costs due to the following practical concerns: (1) the fuel consumption as well the emissions depend on many factors other than the speed, and some factors, such as vessel load and weather conditions, change over the route (for example, the daily bunker consumption for a ship with full load and in ballast could differ by up to $25 - 30\%$ of the total bunker consumption at the same speed [32]); (2) the fuel price varies across ports, so the fuel cost per mile are different even with the same fuel consumption rate; (3) some ships use different types of fuels when sailing inside and outside emission control areas to follow the environmental regulations [6, 14]. Therefore, heterogeneous cost functions in the speed optimization model provide a more accurate assessment of the bunker consumption and emissions.

With no assumption on the cost, the SOOP is a general nonlinear programming problem, so it is difficult to find a global optimal solution in general. In this paper, we assume the cost is a continuously differentiable and strictly convex function of the speed. Many fuel consumption and emission models used in practice satisfy this assumption [31], for example the cubic function frequently used to estimate the daily bunker consumption [34]. Under this assumption, the SOOP can be formulated as a convex optimization problem, and solved by a general convex optimization solver. However, the general solver offers little managerial insight to the ship planner other than the optimal speeds, and does not explore the special structure of the cost functions. Moreover, the SOOP often appears as a subproblem in many more complex models involving other tactical and operational decisions, such as the tramp ship routing and scheduling problem [27], liner service network design [25, 40], and fleet management [1, 17]. Thus an self-contained fast algorithm that can be easily embedded as a subroutine into other exact and heuristic methods is needed for the SOOP. In this paper, we develop a simple and efficient algorithm for the SOOP. The algorithm is able to solve instances of 1,000 nodes in less than one second. The Matlab implementation of the algorithm requires less than 100 lines of code. We also want to mention another observation, that may be of independent interest, is that our algorithm requires much less memory than a general convex optimization solver; it is able to solve instances of 1,000,000 nodes in several hundred seconds while the general solver runs out of memory for instances of 10,000 nodes. We now summarize the contribution of our paper as follows.

- We study a speed optimization problem over a path to minimize the fuel consumption and emissions. The problem features heterogeneous convex costs and speed limits across arcs and service time-windows constraints.

- We develop a simple and efficient algorithm when the cost over each arc is a continuously differentiable and strictly convex function of the speed. Our algorithm is 20 to 100 times faster than a general convex optimization solver on test instances and requires much less memory.

- The solutions found at intermediate steps of our algorithm provide some insights to ship planners on how to balance operating cost and service quality.

The rest of the paper is organized as follows. Section 2 reviews the speed optimization models and algorithms in the literature and how they are integrated into other planning and operational models. Section 3 gives the mathematical description of the problem we study. Sections 4 describes our algorithm and 5 elaborates one critical subroutine of the algorithm. Section 6 illustrates our algorithm with a practical example. Section 7 presents computational results on test instances with our algorithm. We conclude in Section 8.

2 Literature Review

The bunker consumption as well as emissions of a ship depend on many factors including sailing speed, engine technology, and vessel and fuel types. Many functions have been proposed in the literature to quantify the relationship between the daily bunker consumption and sailing speed. The cubic function is frequently used in the literature for the estimation [34, 29, 35]. In [41], the function is assumed to be a power function of the sailing speed, and the power is calculated to be between 2.7 and 3.3 with empirical data, providing some support for the cubic approximation. A quartic function is suggested by [22] when the vessel speed is greater than 20 knots. Vessel-type dependent power functions are used in [10], with degree 3.5 for feeder containerships, degree 4 for medium-sized containerships, and degree 4.5 for jumbo containerships. Another important factor that affect the bunker consumption rate is the vessel load. It is shown in [32] that the difference between bunker consumption for a ship at full load or in ballast at the same speed can be as much as $25 - 30\%$ of the total bunker consumption, for the type of Very Large Crude Carrier. Load-dependent bunker consumption functions have also been considered in [31, 44]. For a comprehensive review of the dependence of bunker consumption and emissions on vessel speed and other factors, we refer the interested readers to [31].

We now review some speed optimization models in maritime transportation. In many models [34, 14], speed over each leg of a route is optimized individually. In practice, speed decisions over different legs of a route are usually interconnected due to constraints like service time-window constraints at different customers. Speed optimization problems with these additional service and scheduling constraints are more challenging. In [13], a speed optimization problem over a fixed route is proposed, with the goal of finding the optimal speed over each leg to minimize fuel emissions while respecting the time-window constraint at each customer. They solve this problem approximately by dicretizing the arrival times and reformulating it as a shortest path problem. When the bunker consumption rate over each arc is the same, an exact algorithm for the speed optimization over a fixed route is proposed in [27] based on divide and conquer. The correctness of this algorithm is proven in [19], under the assumption that the bunker consumption rate over each leg was a smooth nondecreasing convex function of the sailing speed. The algorithm in [27] is later modified in [23] to handle the variable starting time at the route and additional cost components in the objective. Under certain assumptions, the speed optimization problem can also be reformulated as a resource allocation problem with additional constraints [20, 30, 21, 39, 38]. For an excellent review of speed optimization models in maritime transportation, please refer to [32].

Since speed decisions affect transit time, service quality, and operating cost, many maritime transportation models consider jointly optimizing speeds and other decisions, such as fleet size, routing, and scheduling. In [29] and [35], the authors propose to find the optimal speed and the number of vessels deployed together for each route in liner services on the Europe - Far East trade, and investigate how these decisions are affected by the increase of bunker prices. In [25], a long-haul liner service problem is studied to determine the optimal service frequency, containership fleet deployment plan, and sailing speeds over a fixed service route. The problem is formulated as a mixed-integer nonlinear optimization problem, and solved by a tailored branch-and-bound algorithm. In [1], a joint routing, fleet deployment, and speed optimization problem is solved for liner companies. In [17], a short-term fleet deployment problem in liner shipping is studied to find the optimal service frequency and speeds for each vessel type over all the routes, and the problem is solved as a mixed-integer nonlinear optimization problem. In [40], the authors investigate a tactical-level liner ship route schedule design problem by taking into account the time uncertainties at sea and at port, and develop a mixed-integer nonlinear stochastic programming model to minimize the operating cost subject to a required transit time service level. In [16], the authors study the routing and scheduling problem of a heterogeneous tramp fleet over a network to minimize the total operating cost, and use a time based discretization approach to reformulate the problem as a mixed-integer linear programming problem. In [43], a polynomial-time algorithm is developed to solve the containership sailing speed optimization problem. For a more comprehensive review on speed optimization in maritime transportation, please refer to the two excellent surveys [26, 32].

We should mention that speed optimization has also been studied in the context of road transportation, in particular related to green vehicle routing [11, 4]. One such well-studied problem is the pollution-routing problem (PRP) [5], which aims to find the optimal routes and speeds over the legs of each route to minimize the total operating and environmental costs. Heuristic methods [9, 24] and exact algorithms [8, 15] have been proposed to solve different variants of the PRP.

3 Problem description

We now gives a more detailed description of the SOOP. We would like to determine the optimal speed over each leg of a given route, such that the sum of the costs over all legs of the route is minimized and the following constraints are satisfied: 1) Each customer over the route must be served within a prescribed time window (the vehicle will wait if it arrives early); 2) The speed over each leg must satisfy certain speed limits. To give the mathematical formulation of the problem, we first introduce some notations. Let $[m]$ denote the set $\{1, \dots, m\}$ for any given positive integer m. Let $(1, 2, \dots, n+1)$ denote a path or a route with a sequence of $n+1$ nodes. Let $(i, i+1)$ denote the arc from node i to node $i+1$, d_i denote the length of arc $(i, i+1)$, l_i and u_i denote the speed lower and upper limits over arc $(i, i+1)$ respectively, for $i \in [n]$. For each $i \in [n]$, the cost per unit distance is $c_i(v_i)$, assuming the vehicle traverses $(i, i+1)$ at a speed of v_i, so the cost over $(i, i+1)$ is $d_i c_i(v_i)$. We assume $c_i : [0, \infty) \to \mathbb{R}$ is strictly convex and continuously differentiable for each $i \in [n]$. Each node $i \in [n+1]$ must be served within a prescribed time window $[a_i, b_i]$. Let t_i denote the service start time at node i for $i \in [n]$. The goal is to find the values of $v_1, \dots, v_n$ and $t_1, \dots, t_{n+1}$ to minimize the total cost while respecting all the time-window and speed limit constraints. The SOOP can be formulated as follows.

$$\min \sum_{i \in [n]} d_i c_i(v_i) \tag{1a}$$

$$\text{s.t. } t_{i+1} \geq t_i + d_i / v_i, \qquad \forall i \in [n], \tag{1b}$$

$$a_i \leq t_i \leq b_i, \qquad \forall i \in [n+1], \tag{1c}$$

$$l_i \leq v_i \leq u_i, \qquad \forall i \in [n]. \tag{1d}$$

The objective (1a) stipulates that our goal is to minimize the sum of costs over all arcs. Constraints (1b) stipulate that the service start time at node $i+1$ must be no earlier than the sum of the service start time at node i and the travel time over arc $(i, i+1)$. Constraints (1c) and (1d) are the time-window constraints and speed limit constraints, respectively. In [13, 27, 19], the authors focus on solving (1) when the bunker consumption rate c_i's are the same non-decreasing smooth convex function across all arcs. In this paper, we study how to solve (1) when c_i's are different across the arcs.

4 The proposed algorithm

Without loss of generality, we make the following assumptions on the speed optimization problem (1).

- Problem (1) is feasible. This can be easily verified by assigning the maximum possible speed u_i to each arc $(i, i+1)$ and checking if the resulting service start time at each customer i exceeds b_i.

- The time windows $a_1 = b_1 = 0$ and $a_{n+1} = b_{n+1}$. If the last customer has a time window $[a_{n+1}, b_{n+1}]$ such that $a_{n+1} < b_{n+1}$, we can append an artificial customer $n+2$ after customer $n+1$ and set $a_{n+2} = b_{n+2}$ without changing the optimal cost of the original problem. Specifically, we can set the distance $d_{n+1} = 1$, speed limits $l_{n+1} = 1$ and $u_{n+1} = 10$, cost $c_{n+1}(v) = (v-1)^2$, and $a_{n+2} = b_{n+2} = b_{n+1} + 1$. In this way, the vessel leaves customer $n+1$ no later than b_{n+1}, and will always traverse arc $(n+1, n+2)$ at an optimal speed of 1, and serves the artificial customer $n+2$ at time $b_{n+1}+1$, possibly with some waiting. The case with $a_1 < b_1$ can be handled in a similar way. If $a_1 = b_1 > 0$, we can shift all the time windows to the left by an amount of a_1 without changing the optimal objective of the problem.

We also assume that the optimal speed v_i^* that minimizes the function $c_i(v)$ lies within the speed limits for $i \in [n]$. This assumption is satisfied by many speed models in [31].

We first transform the speed optimization problem (1) into an equivalent optimization model, in which the only decision variables are service start time t_i's.

Proposition 1. *The speed optimization problem* (1) *can be reformulated as follows.*

$$\min \sum_{i \in [n]} d_i h_i\left(\frac{d_i}{t_{i+1} - t_i}\right) \tag{2a}$$

$$s.t.\ a_i \le t_i \le b_i, \qquad \forall i \in [n+1], \tag{2b}$$

$$t_{i+1} - t_i \ge d_i/u_i, \qquad \forall i \in [n], \tag{2c}$$

where $h_i : \mathbb{R} \to \mathbb{R}$ and defined as follows

$$h_i(v) = \begin{cases} c_i(v) & \text{if } v \ge v_i^*, \\ c_i(v_i^*) & \text{if } v < v_i^*. \end{cases} \tag{3}$$

Recall that v_i^* is the speed that minimizes $c_i(v)$ where there are no speed limits and $c_i'(v_i^*) = 0$. To prove Proposition 1, we first prove the following proposition.

Proposition 2. *Any optimal solution $(t_0, \ldots, t_n, v_1, \ldots, v_n)$ of problem* (1) *satisfies the following property: for each $i \in [n]$, if $d_i/(t_{i+1} - t_i) \ge v_i^*$, then there is no waiting at customer $i+1$ and $v_i = d_i/(t_{i+1} - t_i)$; if $d_i/(t_{i+1} - t_i) < v_i^*$, then the vessel waits at customer $i+1$ and $v_i = v_i^*$.*

Proof. First consider the case in which $d_i/(t_{i+1} - t_i) \ge v_i^*$. If the vessel waits at customer $i+1$, then $t_{i+1} > t_i + d_i/v_i$. Since $d_i/(t_{i+1} - t_i) \ge v_i^*$, then $t_{i+1} \le t_i + d_i/v_i^*$. Thus $v_i > v_i^*$. We can create a new solution to (1) by replacing the v_i-component in the optimal solution $(t_0, \ldots, t_n, v_1, \ldots, v_i, \ldots, v_n)$ by $v_i' = d_i/(t_{i+1} - t_i)$. Since $v_i' \in [v_i^*, v_i)$, the new solution $(t_0, \ldots, t_n, v_1, \ldots, v_i', \ldots, v_n)$ is still feasible, and cost over arc $(i, i+1)$ is decreased. This contradicts to the optimality of the original solution $(t_0, \ldots, t_n, v_1, \ldots, v_n)$. Therefore, there is no waiting at customer $i+1$ and $v_i = d_i/(t_{i+1} - t_i)$. On the other hand, when $d_i/(t_{i+1} - t_i) < v_i^*$, we have $t_{i+1} > t_i + d_i/v_i^*$ and the vessel waits at customer $i+1$. Since $c_i(v)$ is strictly increasing when $v \ge v_i^*$, it is optimal to travel at speed v_i^*. □

Proof of Proposition 1. We prove that any optimal solution of problem (1) can be transformed to a feasible solution of problem (2) with the same objective value and vice versa. First, given an optimal solution $\{\hat{t}_i\}_{i\in[n+1]}$ and $\{\hat{v}_i\}_{i\in[n]}$ of problem (1), the solution $\{\hat{t}_i\}_{i\in[n+1]}$ satisfies all constraints in (2). Thus $\{\hat{t}_i\}_{i\in[n+1]}$ is a feasible solution of (2). According to Proposition 2, if $d_i/(\hat{t}_{i+1}-\hat{t}_i) \geq v_i^*$, then $\hat{v}_i = d_i/(\hat{t}_{i+1}-\hat{t}_i)$; if $d_i/(\hat{t}_{i+1}-\hat{t}_i) < v_i^*$, then $\hat{v}_i = v_i^*$. In any case, $c_i(\hat{v}_i) = h_i(\frac{d_i}{\hat{t}_{i+1}-\hat{t}_i})$. The objective value of the solution $\{\hat{t}_i\}_{i\in[n+1]}$ in (2) is the same as the objective value of the solution $\{\hat{t}_i\}_{i\in[n+1]}$ and $\{\hat{v}_i\}_{i\in[n]}$ in (1). Conversely, given an optimal solution $\{\hat{t}_i\}_{i\in[n+1]}$ of (2), let $\hat{v}_i = d_i/(\hat{t}_{i+1}-\hat{t}_i)$ if $d_i/(\hat{t}_{i+1}-\hat{t}_i) \geq v_i^*$ and let $\hat{v}_i = v_i^*$ if $d_i/(\hat{t}_{i+1}-\hat{t}_i) < v_i^*$. Then $\{\hat{v}_i\}_{i\in[n]}$ satisfies the speed limit constraints in (1). We also have $h_i(\frac{d_i}{\hat{t}_{i+1}-\hat{t}_i}) = c_i(\hat{v}_i)$ for $i \in [n]$, so the objective value of $\{\hat{t}_i\}_{i\in[n+1]}$ and $\{\hat{v}_i\}_{i\in[n]}$ in (1) is the same as the objective value of $\{\hat{t}_i\}_{i\in[n+1]}$ in (2). $\square$

The transformation in Proposition 1 is inspired by [24] for the speed optimization problem with homogeneous cost. To solve problem (2), we define two related problems. The first is the speed optimization problem from customer s to customer e, which we call SOP(s,e).

$$\begin{aligned} \min \ & \sum_{i=s}^{e-1} d_i h_i\left(\frac{d_i}{t_{i+1}-t_i}\right) \\ \text{s.t. } & t_{i+1}-t_i \geq d_i/u_i, \ i=s,\ldots,e-1, \\ & a_i \leq t_i \leq b_i, \ i=s+1,\ldots,e-1, \end{aligned} \qquad \text{SOP}(s,e)$$

where $t_s = a_s$ and $t_e = a_e$. Our goal is to solve SOP(s,e) for $s=1$ and $e=n+1$. The other problem is the speed optimization problem from customer s to customer e without time-window constraints, which we call SOPR(s,e). The problem SOPR(s,e) is formulated as follows.

$$\begin{aligned} \min \ & \sum_{i=s}^{e-1} d_i h_i\left(\frac{d_i}{t_{i+1}-t_i}\right) \\ \text{s.t. } & t_{i+1}-t_i \geq d_i/u_i, \ i=s,\ldots,e-1, \end{aligned} \qquad \text{SOPR}(s,e)$$

with $t_s = a_s$ and $t_e = a_e$.

We now describe the basic idea of our algorithm to solve SOP(s,e). We first solve the relaxation SOPR(s,e). If the obtained optimal service times satisfy all the time-window constraints, then we stop and output the service times as the optimal solution for SOP(s,e). Otherwise, find a customer K with the maximum time-window violation, fix its service start time to be a_K or b_K (depending on how the time-window constraint at customer K is violated), and divide the original problem into two problems SOP(s,K) and SOP(K,e). Each of the problems SOP(s,K) and SOP(K,e) is solved recursively. Our algorithm is in the same vein as the divide and conquer algorithm proposed in [27] for the SOOP when the cost function $c_i(v)$'s are the same across the arcs, but the procedure and analysis is much more involved due to the heterogeneity of c_i's. The details of our algorithm are described in Algorithm 1.

Theorem 1. *Algorithm 1 solves SOP(s,e) correctly for any $1 \leq s < e \leq n+1$.*

To prove Theorem 1, we first prove the proposition below.

Proposition 3. *Let $(\bar{t}_{s+1},\ldots,\bar{t}_{e-1})$ be a solution obtained by solving SOPR(s,e) with $1 \leq s < e \leq n+1$. Suppose $(\bar{t}_{s+1},\ldots,\bar{t}_{e-1})$ violates at least one time-window constraint of SOP(s,e) and customer K has the maximum time-window violation. Then any optimal solution $(\hat{t}_{s+1},\ldots,\hat{t}_{e-1})$ of SOP(s,e) satisfies $\hat{t}_K = \max\{a_K, \min\{\bar{t}_K, b_K\}\}$.*

Algorithm 1 A recursive algorithm to solve SOP(s, e)

```
 1: function SOP($s, e$)
 2:     if $e - s = 1$ then return $(a_s, a_e)$
 3:     else if $e - s > 1$ then
 4:         Solve SOPR$(s, e)$ and obtain the optimal solution $(\bar{t}_{s+1}, \ldots, \bar{t}_{e-1})$
 5:         if $\bar{t}_i \in [a_i, b_i]$ for $i = s+1, \ldots, e-1$ then return $(a_s, \bar{t}_{s+1}, \ldots, \bar{t}_{e-1}, a_e)$
 6:         else Find customer $K$ with the maximum time-window violation
                $$\delta = \max_{i=s+1,\ldots,e-1} \{a_i - \bar{t}_i, \bar{t}_i - b_i\}$$
 7:             $a_K \leftarrow \max\{a_K, \min\{\bar{t}_K, b_K\}\}$ and $b_K \leftarrow \max\{a_K, \min\{\bar{t}_K, b_K\}\}$
 8:             $(a_s, \bar{t}_{s+1}, \ldots, \bar{t}_{K-1}, a_K) \leftarrow$ SOP$(s, K)$
 9:             $(a_K, \bar{t}_{K+1}, \ldots, \bar{t}_{e-1}, a_e) \leftarrow$ SOP$(K, e)$
10:             return $(a_s, \bar{t}_{s+1}, \ldots, a_K, \ldots, \bar{t}_{e-1}, a_e)$
```

Proof. Customer K has either the largest tardiness or the largest earliness. Below we prove for the case when customer K has the largest tardiness. The case when customer K has the largest earliness can be proven analogously.

We have $\bar{t}_K > b_k$. Our goal is to prove that any optimal solution $(\hat{t}_{s+1}, \ldots, \hat{t}_{e-1})$ of SOP(s, e) satisfies $\hat{t}_K = b_K$. Suppose $\hat{t}_K < b_K$. Let $I = \arg\max\{i \mid \hat{t}_i = b_i, s \le i < K\}$ and $J = \arg\min\{i \mid t_i = \hat{b}_i, K < i \le e\}$. We claim that there exist integers p and q such that $I \le p < K < q \le J - 1$ and

$$\frac{d_p}{\bar{t}_{p+1} - \bar{t}_p} < \frac{d_p}{\hat{t}_{p+1} - \hat{t}_p}, \tag{4a}$$

$$\frac{d_q}{\bar{t}_{q+1} - \bar{t}_q} > \frac{d_q}{\hat{t}_{q+1} - \hat{t}_q}. \tag{4b}$$

To see this, since customer K has the maximum time-window violation, $\bar{t}_K - b_K \ge \bar{t}_I - b_I$. Then $\bar{t}_K - \bar{t}_I \ge b_K - b_I$. Meanwhile, $\hat{t}_K < b_K$ and $\hat{t}_I = b_I$. Thus $\bar{t}_K - \bar{t}_I > \hat{t}_K - \hat{t}_I$. Since $\bar{t}_K - \bar{t}_I = \sum_{i=I}^{K-1} (\bar{t}_{i+1} - \bar{t}_i)$ and $\hat{t}_K - \hat{t}_I = \sum_{i=I}^{K-1} (\hat{t}_{i+1} - \hat{t}_i)$, there must exist p such that $I \le p < K$ and $\bar{t}_{p+1} - \bar{t}_p > \hat{t}_{p+1} - \hat{t}_p$. Thus (4a) holds. Similarly, we can prove that there must exist q such that $K < q \le J - 1$ and $\bar{t}_{q+1} - \bar{t}_q < \hat{t}_{q+1} - \hat{t}_q$. Thus (4b) holds.

The vector $(\bar{t}_{s+1}, \ldots, \bar{t}_{e-1})$ is an optimal solution of SOPR(s, e). Without loss of generality, we can assume that $\frac{d_i}{\bar{t}_{i+1} - \bar{t}_i} \ge v_i^*$ for $i = s, \ldots, e-2$, since it's always optimal to travel over each arc at speed no smaller than v_i^* and there is no need to wait at any customer $i \le e - 1$. (There might be some waiting at the last customer e if a_e is very large.) Consider the KKT conditions of SOPR(s, e).

$$H_i(\frac{d_i}{t_{i+1} - t_i}) + \nu_i = H_{i-1}(\frac{d_{i-1}}{t_i - t_{i-1}}) + \nu_{i-1}, i = s+1, \ldots, e-1, \tag{5a}$$

$$t_{i+1} - t_i \ge d_i / u_i, i = s, \ldots, e-1 \tag{5b}$$

$$\nu_i (t_{i+1} - t_i - d_i / u_i) = 0, i = s, \ldots, e-1, \tag{5c}$$

$$\nu_i \ge 0, i = s, \ldots, e-1, \tag{5d}$$

where $H_i : [l_i, u_i] \to \mathbb{R}$ is defined as $H_i(x) = x^2 h_i'(x)$. It is easy to verify that H_i is nondecreasing over $[l_i, u_i]$ and strictly increasing over $[v_i^*, u_i]$ for each $s \le i \le e - 1$.

Since the constraints in SOPR(s,e) are linear, we can find dual variables $(\bar{\nu}_s,\dots,\bar{\nu}_{e-1})$ associated with the optimal solution $(\bar{t}_{s+1},\dots,\bar{t}_{e-1})$ such that $(\bar{t}_{s+1},\dots,\bar{t}_{e-1})$ and $(\bar{\nu}_s,\dots,\bar{\nu}_{e-1})$ satisfy (5). By (4a), $\frac{d_p}{\bar{t}_{p+1}-\bar{t}_p} < \frac{d_p}{\hat{t}_{p+1}-\hat{t}_p} \le u_i$. Then $\bar{\nu}_p = 0$ according to (5c). By (5a) we have $H_p(\frac{d_p}{\bar{t}_{p+1}-\bar{t}_p}) = H_q(\frac{d_q}{\bar{t}_{q+1}-\bar{t}_q}) + \bar{\nu}_q$, which implies that $H_p(\frac{d_p}{\bar{t}_{p+1}-\bar{t}_p}) \ge H_q(\frac{d_q}{\bar{t}_{q+1}-\bar{t}_q})$. Since H_p is strictly increasing over $[v_p^*,u_p]$, then by (4a),

$$H_p(\frac{d_p}{\bar{t}_{p+1}-\bar{t}_p}) < H_p(\frac{d_p}{\hat{t}_{p+1}-\hat{t}_p}).$$

By (4b), we have

$$H_q(\frac{d_q}{\bar{t}_{q+1}-\bar{t}_q}) \ge H_q(\frac{d_q}{\hat{t}_{q+1}-\hat{t}_q}).$$

Then

$$H_p(\frac{d_p}{\hat{t}_{p+1}-\hat{t}_p}) > H_q(\frac{d_q}{\hat{t}_{q+1}-\hat{t}_q}). \tag{6}$$

Similarly, since $(\hat{t}_{s+1},\dots,\hat{t}_{e-1})$ is an optimal solution of SOP(s,e), we can find associated dual variables $(\hat{\lambda}_{s+1},\dots,\hat{\lambda}_{e-1})$, $(\hat{\mu}_{s+1},\dots,\hat{\mu}_{e-1})$, and $(\hat{\nu}_s,\dots,\hat{\nu}_{e-1})$ such that they satisfy the KKT conditions of SOP(s,e).

$$a_i \le t_i \le b_i, i = s+1,\dots,e-1, \tag{7a}$$
$$t_{i+1}-t_i \ge d_i/u_i, i = s,\dots,e-1 \tag{7b}$$
$$\lambda_i \ge 0, i = s+1,\dots,e-1, \tag{7c}$$
$$\mu_i \ge 0, i = s+1,\dots,e-1, \tag{7d}$$
$$\nu_i \ge 0, i = s,\dots,e-1, \tag{7e}$$
$$\lambda_i(b_i - t_i) = 0, i = s+1,\dots,e-1, \tag{7f}$$
$$\mu_i(t_i - a_i) = 0, i = s+1,\dots,e-1, \tag{7g}$$
$$\nu_i(t_{i+1}-t_i-d_i/u_i) = 0, i = s,\dots,e-1, \tag{7h}$$
$$H_i(\frac{d_i}{t_{i+1}-t_i}) - H_{i-1}(\frac{d_{i-1}}{t_i-t_{i-1}}) + \lambda_i - \mu_i + \nu_i - \nu_{i-1} = 0, i = s+1,\dots,e-1. \tag{7i}$$

By (4b), we have $u_q \ge \frac{d_q}{\bar{t}_{q+1}-\bar{t}_q} > \frac{d_q}{\hat{t}_{q+1}-\hat{t}_q}$. According to (7h), $\hat{\nu}_q = 0$. Since $\hat{t}_i < b_i$ for $I < i < J$, $\bar{\lambda}_i = 0$ by (7f). Then according to (7i),

$$H_i(\frac{d_i}{\hat{t}_{i+1}-\hat{t}_i}) + \hat{\nu}_i = H_{i-1}(\frac{d_{i-1}}{\hat{t}_i-\hat{t}_{i-1}}) + \hat{\nu}_{i-1} + \mu_i \ge H_{i-1}(\frac{d_{i-1}}{\hat{t}_i-\hat{t}_{i-1}}) + \hat{\nu}_{i-1},$$

for any $I < i < J$. Then $H_q(\frac{d_q}{\hat{t}_{q+1}-\hat{t}_q}) + \hat{v}_q \ge H_p(\frac{d_p}{\hat{t}_{p+1}-\hat{t}_p}) + \hat{v}_p$. Combining with the fact that $\hat{\nu}_q = 0$, we have

$$H_q(\frac{d_q}{\hat{t}_{q+1}-\hat{t}_q}) = H_q(\frac{d_q}{\hat{t}_{q+1}-\hat{t}_q}) + \hat{v}_q \ge H_p(\frac{d_p}{\hat{t}_{p+1}-\hat{t}_p}) + \hat{v}_p \ge H_p(\frac{d_p}{\hat{t}_{p+1}-\hat{t}_p}).$$

The above inequality contradicts (6). □

We are now ready to prove Theorem 1.

Proof of Theorem 1. We prove by induction on $|e-s|$. For the base case SOP(s,e) with $|e-s|=1$, Step 2 in Algorithm 1 correctly outputs the trivial optimal solution (a_s, a_e). Now assume that Algorithm 1 correctly solves SOP(s,e) with $|e-s| \leq k$ for some integer $k \geq 1$. We want to show that Algorithm 1 correctly solves SOP(s,e) with $|e-s| = k+1$. There are two cases to consider. In the first case, Algorithm 1 goes to Step 5. The solution $(a_s, \bar{t}_{s+1}, \dots, \bar{t}_{e-1}, a_e)$ is optimal for SOPR(s,e), and satisfies all time-window constraints in SOP(s,e), so it is also optimal for SOP(s,e). In the second case, Algorithm 1 goes to Step 6. By Proposition 3, any optimal solution $(\hat{t}_{s+1}, \dots, \hat{t}_{e-1})$ of SOP(s,e) satisfies $\hat{t}_K = \bar{t}_K$. Then SOP(s,e) is divided into two subproblems SOP(s,K) and SOP(K,e). By the induction hypothesis, Algorithm 1 correctly output optimal solutions for these two subproblems. Combining the two solutions gives an optimal solution of SOP(s,e) with $|e-s| = k+1$. □

Theorem 1 also implies the following result, which guarantees that each subproblem we solve in Algorithm 1 will be feasible.

Corollary 1. *If the original problem SOP$(1, n+1)$ is feasible, then each subproblem SOPR(s,e) encountered in Algorithm 1 is also feasible.*

In the next section, we explore in detail how to solve the subproblem SOPR(s,e) for $1 \leq s < e \leq n+1$.

5 An algorithm to solve SOPR(s,e)

We describe in detail Step 4 in Algorithm 1, i.e., how to find a solution to SOPR(s,e). We first present the formulation of SOPR(s,e) again.

$$\min \sum_{i=s}^{e-1} d_i h_i\left(\frac{d_i}{t_{i+1}-t_i}\right) \tag{8a}$$

$$\text{s.t. } t_{i+1} - t_i \geq d_i/u_i, \ i = s, \dots, e-1, \tag{8b}$$

$$t_s = a_s, t_e = a_e. \tag{8c}$$

The problem SOPR(s,e) is essentially a resource allocation problem with separable convex objective functions [20, 30]. To see this, define $x_i = t_{i+1} - t_i$ for $s \leq i \leq e-1$. Note that the service time t_s and t_e are fixed to be a_s and a_e, respectively. Thus $\sum_{i=s}^{e-1} x_i = \sum_{i=s}^{e-1}(t_{i+1}-t_i) = a_e - a_s$. We reformulate SOPR$(s,e)$ as follows.

$$\min \sum_{i=s}^{e-1} f_i(x_i) \tag{9a}$$

$$\text{s.t. } \sum_{i=s}^{e-1} x_i = a_e - a_s, \tag{9b}$$

$$x_i \geq \frac{d_i}{u_i}, i = s, \dots, e-1, \tag{9c}$$

$$\tag{9d}$$

where $f_i : (0, \infty) \to \mathbb{R}$ is defined as follows: $f_i(x_i) = d_i h_i(\frac{d_i}{x_i})$.

Proposition 4. *The function f_i is convex over $(0,\infty)$. Its derivative $f_i'(x_i)$ is strictly increasing when $0 < x_i \leq \frac{d_i}{v_i^*}$ and $f_i'(x_i) = 0$ when $x_i \geq \frac{d_i}{v_i^*}$.*

Proof. The derivative $f_i'(x_i) = -h_i'(\frac{d_i}{x_i})\frac{d_i^2}{(x_i)^2} = -c_i'(\frac{d_i}{x_i})\frac{d_i^2}{x_i^2}$ when $0 < x_i < d_i/v_i^*$. Since c_i is strictly convex and increasing over (v_i^*, ∞), $c_i'(\frac{d_i}{x_i})$ is strictly decreasing and positive when $x_i \in (0, \frac{d_i}{v_i^*})$. Then the derivative $f_i'(x_i)$ is negative and strictly increasing when $x_i \in (0, \frac{d_i}{v_i^*})$. In particular, $f_i'(x_i) = 0$ when $x_i = \frac{d_i}{v_i^*}$. On the other hand, $f_i'(x_i) = 0$ when $x_i \geq \frac{d_i}{v_i^*}$. Thus $f_i'(x_i)$ is non-decreasing over $(0, \infty)$ and $f_i(x_i)$ is convex. □

Many algorithms have been developed to solve the resource allocation problem with continuous variables [20, 30]. Note that the function f_i's in our problem are not strictly convex according to Proposition 4, so efficient algorithms such as the one proposed in [45] cannot be applied. Below we explore the property of f_i's and present an efficient algorithm to solve (9).

Since problem (9) is a convex optimization problem with linear constraints, any optimal solution of (9) satisfies the following KKT conditions.

$$f_i'(x_i) - \alpha - \nu_i = 0, i = s, \ldots, e-1, \tag{10a}$$

$$\sum_{i=s}^{e-1} x_i = a_e - a_s, \tag{10b}$$

$$x_i \geq \frac{d_i}{u_i}, i = s, \ldots, e-1, \tag{10c}$$

$$\nu_i(\frac{d_i}{u_i} - x_i) = 0, i = s, \ldots, e-1, \tag{10d}$$

$$\nu_i \geq 0, i = s, \ldots, e-1, \tag{10e}$$

where α and ν_i's are the Lagragian multiplier associated with constraint (9b) and (9c), respectively. Eliminating variable ν_i's from (10), we obtain the following conditions.

$$\sum_{i=s}^{e-1} x_i = a_e - a_s, \tag{11a}$$

$$x_i \geq \frac{d_i}{u_i}, i = s, \ldots, e-1, \tag{11b}$$

$$(f_i'(x_i) - \alpha)(\frac{d_i}{u_i} - x_i) = 0, i = s, \ldots, e-1, \tag{11c}$$

$$f_i'(x_i) - \alpha \geq 0, i = s, \ldots, e-1, \tag{11d}$$

In the rest of this section, we focus on solving (11). Since the derivative $f_i'(x_i)$ is strictly increasing and continuous when $\frac{d_i}{u_i} \leq x_i < \frac{d_i}{v_i^*}$, there exists the inverse of $f_i'(x_i)$, which we denote p_i. That is $p_i(y) = x_i$ implies $y = f_i'(x_i)$. The function p_i is also strictly increasing and continuous over $[f_i'(\frac{d_i}{u_i}), 0)$. Let $L = \min_i\{f_i'(\frac{d_i}{u_i})\} = -\max_i\{c_i'(u_i)u_i^2\}$. For each $i = s, \ldots, e-1$, define $g_i : [L, 0) \to \mathbb{R}$ as follows:

$$g_i(\alpha) = \begin{cases} p_i(\alpha), & f_i'(\frac{d_i}{u_i}) \leq \alpha < 0 \\ \frac{d_i}{u_i}, & L \leq \alpha < f_i'(\frac{d_i}{u_i}). \end{cases}$$

Proposition 5.

1. *There exists a solution x and α to (11) such that $\alpha \in [L, 0]$.*

2. *The KKT system (11) has a solution (x, α) with $\alpha = 0$ if and only if $\sum_{i=s}^{e-1} \frac{d_i}{v_i^*} \leq a_e - a_s$.*

3. *Given* $\alpha \in [L, 0)$, $x_i = g_i(\alpha)$ *is the only solution that satisfies* (11b) *to* (11d). *The function* $T_{se}(\alpha) = \sum_{i=s}^{e-1} g_i(\alpha)$ *is continuous and strictly increasing over* $[L, 0)$.

Proof.

1. First $\alpha \leq 0$, since $\alpha \leq f_i'(x_i)$ according to (11d) and $f_i'(x_i) \leq 0$ for any i. Suppose there exists a solution x and α to (11) such that $\alpha < L$. Since $f_i'(x_i)$ is strictly increasing over $(0, \frac{d_i}{v_i^*})$ and $x_i \geq \frac{d_i}{u_i}$, $f_i'(x_i) \geq f_i'(\frac{d_i}{u_i}) \geq L > \alpha$ for any i. Thus $\frac{d_i}{u_i} - x_i = 0$ according to (11c). It is not difficult to verify that x and $\alpha = L$ is also feasible for (11).

2. If x and $\alpha = 0$ satisfy (11), then $f_i'(x_i) \geq \alpha = 0$ according to (11d). Since $f_i'(x_i) \leq 0$ as well, $f_i'(x_i) = 0$ for any i. Then $x_i \geq \frac{d_i}{v_i^*}$ for each i. According to (11a), $\sum_{i=s}^{e-1} x_i = a_e - a_s$, so $\sum_{i=s}^{e-1} \frac{d_i}{v_i^*} \leq a_e - a_s$. Conversely, if $\sum_{i=s}^{e-1} \frac{d_i}{v_i^*} \leq a_e - a_s$, then the solution $x_i = \frac{d_i}{v_i^*}$ for $s \leq i \leq e-2$, $x_{e-1} = a_e - a_s - \sum_{i=s}^{e-2} \frac{d_i}{v_i^*}$, and $\alpha = 0$ satisfy the KKT system (11).

3. If $\alpha < f_i'(\frac{d_i}{u_i})$, then $f_i'(x_i) \geq f_i'(\frac{d_i}{u_i}) > \alpha$. Thus $\frac{d_i}{u_i} - x_i = 0$ according to (11c), and $x_i = \frac{d_i}{u_i} = g_i(\alpha)$. If $f_i'(\frac{d_i}{u_i}) < \alpha < 0$, then by (11d) $f_i'(x_i) \geq \alpha > f_i'(\frac{d_i}{u_i})$. Then $x_i > \frac{d_i}{u_i}$. Thus $f_i'(x_i) = \alpha$ according to (11c) and $x_i = p_i(\alpha) = g_i(\alpha)$. If $\alpha = f_i'(\frac{d_i}{u_i})$, from (11c) we have either $x_i = \frac{d_i}{u_i}$ or $f_i'(x_i) = \alpha = f_i'(\frac{d_i}{u_i})$. In either case, $x_i = g_i(\alpha)$. Moreover, $g_i(\alpha)$ is continuous over $[L, 0)$ and strictly increasing over $[f_i'(\frac{d_i}{u_i}), 0)$ for each i, so $T_{se}(\alpha)$ is continuous and strictly increasing over $[L, 0)$.

$\square$

According to Proposition 5, we can solve (11) in the following two steps.

1. Check if $\sum_{i=s}^{e-1} \frac{d_i}{v_i^*} \leq a_e - a_s$. If so, $x_i = \frac{d_i}{v_i^*}$ for $s \leq i \leq e-2$, $x_{e-1} = a_e - a_s - \sum_{i=s}^{e-2} \frac{d_i}{v_i^*}$, and $\alpha = 0$ form a solution to (11). Otherwise go to next step.

2. Find an $\alpha \in [L, 0)$ such that $T_{se}(\alpha) = a_e - a_s$. Then $x_i = g_i(\alpha)$ for each i and α form a solution to (11).

If closed forms of g_i's and the inverse of T_{se} is available, then the value of α and x_i's can be computed through a formula in Step 2. Otherwise, we do a binary search in the interval $[L, 0)$ to find the appropriate α and associated x_i's.

6 A case study with our algorithm

In this section, we illustrate our algorithm with a practical example. Consider a general cargo ship on an Asia-North Europe route, which starts from Shanghai and ends at Rotterdam with five ports of call in between. The locations of the ports are shown in Figure 1. The sequence of ports to visit, along with corresponding service time windows, demands, and distances between ports are shown in Table 1. The distances between ports are calculated based on the data from `https://sea-distances.org/`. Each port on the route, except the origin and the destination, has a time window of five days. We would like to find a sailing speed profile to minimize the total bunker consumption while respecting the time-window constraint at each port.

We estimate the bunker consumption over each leg in the following way: suppose F_0 is the daily bunker consumption of the ship at a nominal speed v_0 (say 15 knots) and load q_0. The daily bunker consumption of a ship is approximately proportional to the third power of it sailing speed [34] and

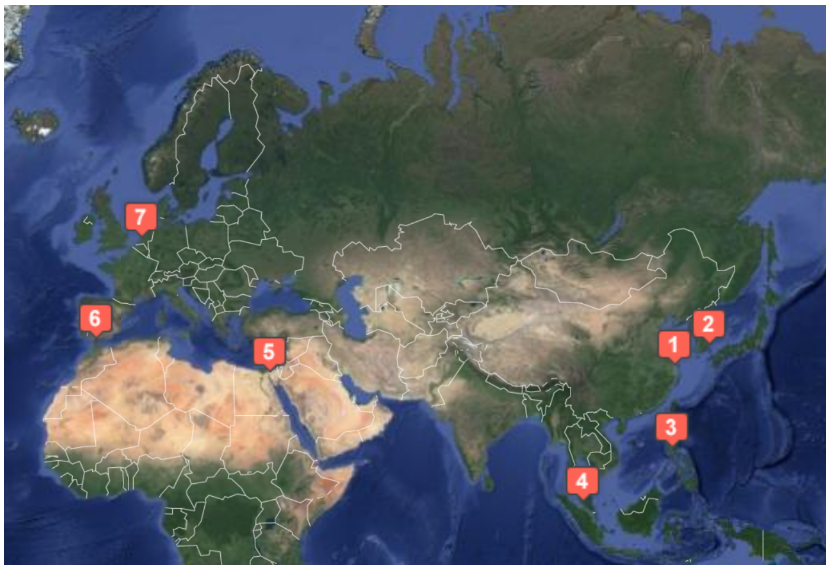

Figure 1: The locations of all the ports

	Port	Cargo to unload (tons)	Time window (hour)	Distance (nautical miles)
1	Shanghai	0	–	428
2	Busan	1959	$[18, 138]$	1226
3	Manila	2167	$[99, 219]$	1166
4	Singapore	2214	$[186, 306]$	4282
5	Suez	2202	$[438, 558]$	1740
6	Algeciras	1086	$[562, 682]$	1187
7	Rotterdam	2915	$[749, 749]$	–

Table 1: Data for an Asia-North Europe route

proportional to the ship load [31]. So for a ship with a load q sailing at speed v, it daily bunker consumption is approximately $F_0 \frac{q}{q_0}(\frac{v}{v_0})^3$. Then the total bunker consumption over leg $(i, i+1)$ is $\frac{F_0}{q_0 v_0^3} \cdot d_i q v^2$. Note that the coefficient $\frac{F_0}{q_0 v_0^3}$ is not significant here, since it appears in the cost over every arc and does not affect the optimal speeds. The speed upper limit over each leg is 20 knots.

Our algorithm solves this instance after solving three SOPR subproblems, SOPR(1,7), SOPR(1,5), and SOPR(5,7). The optimal service start time at each port by solving SOPR(1,7) is given at the second row of Table 2. Notice that the time-window constraint at Suez is violated. Following our algorithm and Proposition 3, the final optimal service start time at Suez should be 558, the right endpoint of its time window. Then we fix the service start time at Suez to 558, and solve two subproblem SOPR(1,5) and SOPR(5,7). The optimal service start times of these two problems are given at the third and four rows of Table 2, respectively. No time-window constraints are violated this time, so the current service start time at each port is optimal for the original problem. The optimal sailing speed from each port to the next port can be calculated according to Proposition 2, and is given at the fifth row of Table 2.

From the solution procedure of our algorithm, it can be seen that the bottleneck to achieve a fuel-efficient speed profile is at Suez. If the service time-window at Suez is flexible and can be changed to $[450, 570]$ (between 18.75 and 23.75 days leaving Shanghai), then the optimal service start times by solving SOPR$(1, 7)$ (the second row of Table 2) will already be optimal and we

Service starting time (hour)	Shanghai	Busan	Manila	Singapore	Suez	Algeciras	Rotterdam
SOPR(1,7)	0	40.5	150.2	246.8	*567.3*	679.9	749
SOPR(1,5)	0	39.8	147.7	242.7	**558**		
SOPR(5,7)					**558**	676.4	749
Optimal speeds (knots)	10.7	11.4	12.3	13.6	14.7	16.3	

Table 2: Service start times and optimal speeds given by our algorithm

can achieve a 0.24% reduction on the total fuel consumption. On the other hand, if the late time window in Algeciras is 672 hours (28 days) after leaving Shanghai, then the time-window constraint at Algeciras becomes the second bottleneck. The optimal service start time at Algeciras will be fixed to 672 hours after leaving Shanghai, and the total fuel cost will increase by 0.14%. These changes in cost will become significant when a yearly schedule with many ship routes are taken into account, especially with a tight profit margin in current shipping industry. Therefore the solutions at the intermediate steps of our algorithm provide some insights and guidelines for ship planners on how to balance operating cost and service quality.

7 Computational experiments

We now test our algorithm on two set of instances, in the context of maritime and road transportation respectively. We then compare the performance of our algorithm with that of CVX, a popular Matlab package for specifying and solving convex programs [7, 18]. We use CVX's default solver SDPT3 [36]. To make a fair comparison, we implement our algorithm in Matlab. In our algorithm, we set the numerical accuracy in each binary search to be 10^{-5}. All computational experiments are performed on a laptop with an Intel i7 processor at 2.20 GHz and 16 GB memory running Windows 10. All test instances mentioned below can be downloaded at `http://www.menet.umn.edu/~qhe/`.

7.1 Maritime Transportation

7.1.1 Instance generation

We use the empirical function $c_{\text{mt}}(v) = 0.0036v^2 - 0.1015v + 0.8848$ in [13] as the basis to generate heterogeneous fuel consumption functions over all the arcs. The function $c_{\text{mt}}(v)$ is obtained based on the real fuel consumption and sailing speed data for a particular ship. In our instances, the fuel consumption rate (ton per nautical mile) is $\beta_2 v^2 + \beta_1 v + 0.8848$, where β_2 and β_1 are independently drawn from the uniform distributions over $[0.0036 - 0.0001, 0.0036 + 0.0001]$ and $[-0.1015 - 0.005, -0.1015 + 0.005]$, respectively. We set the lower and upper speed limits over each arc to 0 and 25 knots, respectively. The distance between two consecutive customers is uniformly drawn from any integers between 100 and 1000 (nautical miles). To generate the time windows, we compute the arrival time at customer i, denoted by A_i, when the ship sails at 20 knots over each arc. Then the early time window a_i of each customer i is a random scalar uniformly drawn from $[A_i - 20, A_i]$, and the late time window $b_i = a_i + 240$. So the width of each time window at each customer is 240 hours or 10 days. We generate 10 instances for each $n \in \{10, 100, 1000, 5000\}$, where n is the total number of nodes on the path.

7.1.2 Computational results

We compare the performance of our algorithm with that of CVX/SDPT3. The results on 5000-customer instances are presented in Table 3. Note that CVX is not able to solve instances of 10,000 nodes due to insufficient memory.

Instance	Our algorithm		CVX/SDPT3	
	Objective	Time(s)	Objective	Time(s)
1	815930	3.74	815930	65.88
2	794250	3.60	794250	65.59
3	806890	3.56	807190	63.92
4	812280	2.53	812280	65.86
5	807310	3.76	807310	64.98
6	802350	3.65	802350	65.17
7	796750	2.56	796750	64.19
8	803330	3.75	803550	62.00
9	796790	3.65	796790	65.27
10	806500	2.56	806500	60.94

Table 3: Comparison of our algorithm and CVX/SDPT3 on the maritime instances with 5000 customers.

It can be seen that our algorithm is around 20 times faster than CVX/SDPT3 on average. Our algorithm also finds a slightly better solution for instance 3 (0.04% improvement) and instance 8 (0.03% improvement), respectively. CVX/SDPT3 is not able to solve these two instances to optimality accurately (the returned status is "`Inaccurate/Solved`"). We should also mention that our algorithm indeed achieves high accuracy for all the instances: each optimal solution found by our algorithm is feasible; on the other hand, each solution found by CVX slightly violates the time-window constraints (by a small amount of 10^{-14} to 10^{-8}), due to the infeasible path-following algorithm used by SDPT3. The detailed results of the two algorithms for instances with 10, 100, and 1000 customers are provided in Table 7 in the appendix. The optimal costs found for these smaller instances are the same by both algorithms.

We summarize the average running time of the two algorithms for instances of different sizes. We also test more instances to further investigate how the running time of each algorithm grows

Instance size		10	100	1000	5000
Running Time(s)	Our algorithm	1.02×10^{-4}	2.78×10^{-2}	0.44	3.34
	CVX/SDPT3	0.84	1.59	9.69	64.38

Table 4: Running times of our algorithm and CVX/SDPT3 on the maritime instances.

with the size of the instance. We generate 20 instances for each $n = 10k, 100k, n = 1000k$ for $1 \leq k \leq 10$. The average running time of the two algorithms for different-sized instances are shown in Figure 2. The growth rate of running time are similar for both algorithms for large n. For instances of the same size, the average running time of our algorithm is approximately one order of magnitude smaller than that of CVX/SDPT3 on the instances of the same size.

We also want to mention one observation that may be of independent interest is that our algorithm requires much less memory than CVX/SDPT3. CVX/SDPT3 runs out of memory with

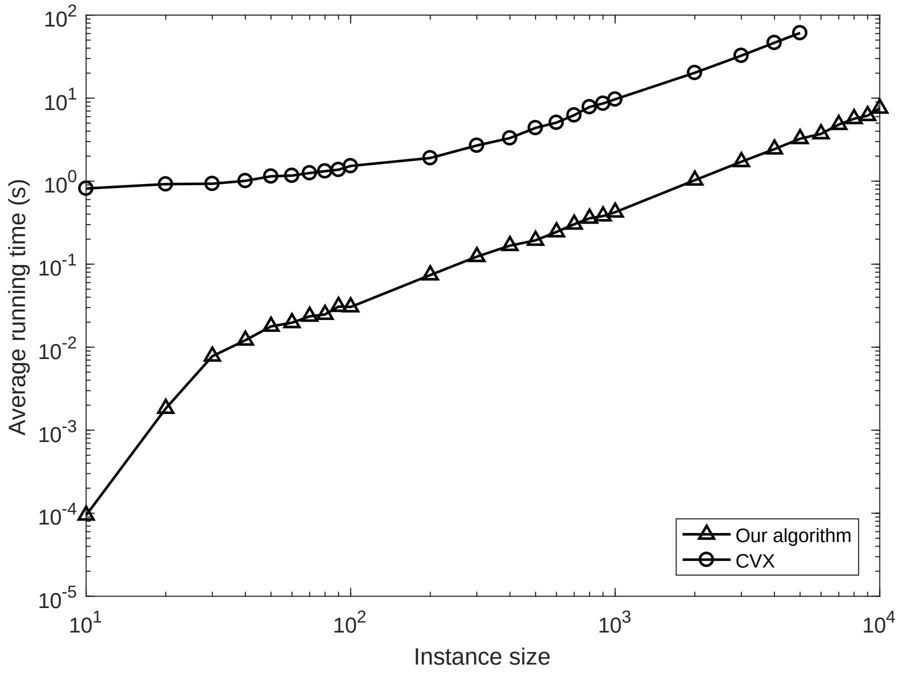

Figure 2: Average running time of our algorithm and CVX/SDPT3 versus the total number of nodes

instances of 10,000 nodes, while our algorithm is able to solve instances of 1,000,000 nodes in several hundred seconds.

7.2 Road transportation

In this section, we test our algorithm on instances in road transportation. Given a sequence of nodes to visit over a route, the goal is to determine the truck speed over each leg of the route to minimizing the total greenhouse gas emissions.

7.2.1 Instance generation

We use the Comprehensive Modal Emission Model (CMEM) developed by the Center of Environmental Research & Technology at the University of California at Riverside [2, 3] as the basis of our emission function over each arc. The pollution-routing problem [5] has used the CMEM for one type of heavy-duty trucks in their objective function to minimize the total cost including the emission cost. Specifically, the emission function in the CMEM is $c(v) = \gamma_1 v^2 + \gamma_2 v^{-1}$ with $\gamma_1 = 1.412 \times 10^{-7}$ and $\gamma_2 = 1.018 \times 10^{-3}$ [5, 24] (we only consider the speed-dependent components in the CMEM since other components do not affect the optimal speeds). In our instances, the emission function over each arc is $\beta_1 v^2 + \beta_2 v^{-1}$, where β_1 and β_2 are random scalars independently drawn from the normal distribution $N(\gamma_1, (0.2 \times 10^{-7})^2)$ and $N(\gamma_2, (0.2 \times 10^{-3})^2)$, respectively. We set the lower and upper speed limits over each arc to 0 and 60 miles per hour, respectively. The distance between two consecutive customers is uniformly drawn from any integers between 40 and 240 (miles). To generate the time windows, we compute the arrival time at customer i, denoted by A_i, when the truck travels at 80% of the maximum speed over each arc. Then the early time window a_i of each customer i is a random scalar uniformly drawn from $[A_i - 0.5, A_i]$, and the late time window $b_i = a_i + 1$. We generate four groups of instances with a total number of 10, 100, 1000, and 5000 nodes respectively, each group with 10 instances.

7.2.2 Computational results

We compare the performance of our algorithm with that of CVX/SDPT3. The results on 5000-node instances are presented in Table 5.

Instance	Our algorithm		CVX/SDPT3	
	Objective	Time(s)	Objective	Time(s)
1	279.44	1.05	279.44	102.23
2	278.82	1.03	278.82	102.92
3	280.46	1.11	280.46	102.47
4	278.78	1.07	278.78	113.66
5	277.55	1.06	277.55	111.97
6	276.68	1.10	276.68	103.31
7	280.50	1.07	280.50	106.00
8	278.33	1.04	278.33	102.56
9	279.82	1.07	279.82	108.95
10	277.79	1.11	277.79	104.22

Table 5: Comparison of our algorithm and CVX/SDPT3 on the road transportation instances with 5000 nodes.

The optimal costs given by both algorithms for each instance are the same, but our algorithm is around 100 times faster than CVX/SDPT3. The detailed results of the two algorithms for instances with 10, 100, and 1000 customers are provided in Table 8 in the appendix. The optimal costs found by both algorithms for each instance turn out to be the same. We summarize the average running time of the two algorithms for instances of different sizes.

Instance size		10	100	1000	5000
Running Time(s)	Our algorithm	4.29×10^{-4}	7.00×10^{-3}	0.14	1.07
	CVX/SDPT3	0.74	1.78	14.95	101.16

Table 6: Running times of our algorithm and CVX/SDPT3 on the road transportation instances.

We also test more instances to investigate how the running time of both algorithms grows with the size of the instance. We generate 20 instances for each $n = 10k, 100k, 1000k$ for $1 \le k \le 10$. The average running time of the two algorithms for different-sized instances are shown in Figure 3. The grow rates of running time as the instance size increases are similar for both algorithms for large n. For instances of the same size, the average running time of our algorithm is approximately two orders of magnitude smaller than that of CVX/SDPT3.

8 Conclusion

In this paper, we study a speed optimization problem over a path to minimize the fuel cost and emissions, in which the cost is assumed to be a continuously differentiable and strictly convex function of the speed and different across the arcs. We develop a fast and efficient algorithm that is above to solve instances of 10,000 nodes in several seconds. The solutions found at the

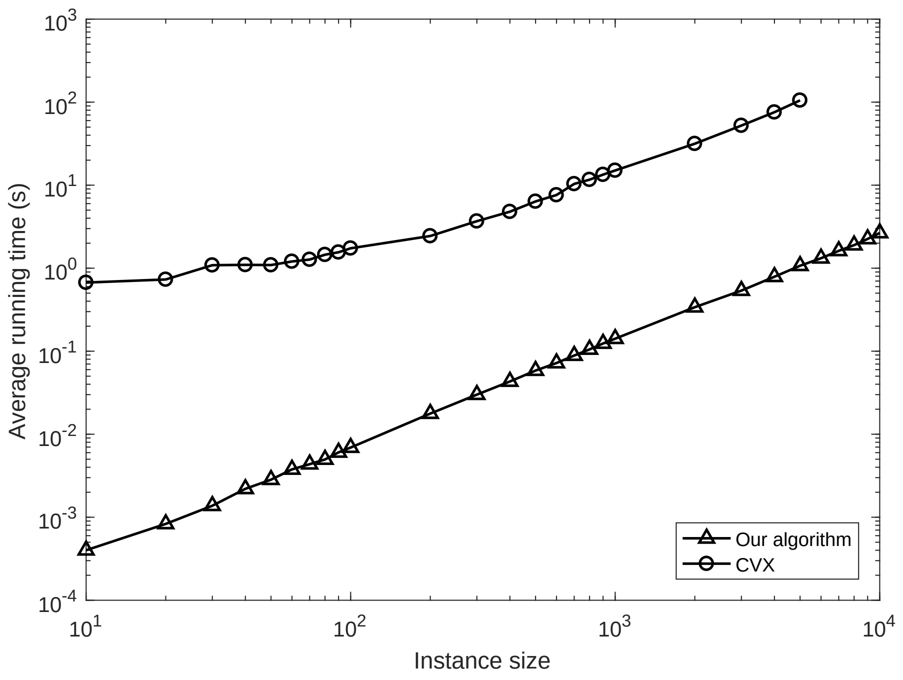

Figure 3: Average running time of our algorithm and CVX/SDPT3 versus the total number of nodes

intermediate steps of our algorithm also provide some insights for ship planners to investigate optimal speed profiles under different scenarios of service time windows, thus achieving a better understanding of how to balance operating cost and service quality. Since speed decision is usually considered along with other tactical and operational decisions in ship planning, speed optimization over a path usually appears as a subproblem in many problems. We hope that our algorithm can be used as an efficient subroutine in heuristic and exact algorithms for more complex planning and operational models.

References

[1] J. F. Alvarez. Joint routing and deployment of a fleet of container vessels. *Maritime Economics & Logistics*, 11(2):186–208, 2009.

[2] M. Barth, F. An, T. Younglove, G. Scora, C. Levine, M. Ross, and T. Wenzel. Development of a comprehensive modal emissions model. Technical report, National Cooperative Highway Research Program, Transportation Research Board, 2000.

[3] M. Barth, G. Scora, and T. Younglove. Modal emissions model for heavy-duty diesel vehicles. *Transportation Research Record*, 1880(1):10–20, 2004.

[4] T. Bektaş, E. Demir, and G. Laporte. Green vehicle routing. In N. H. Psaraftis, editor, *Green Transportation Logistics*, pages 243–265. Springer International Publishing, 2016.

[5] T. Bektaş and G. Laporte. The pollution-routing problem. *Transportation Research Part B: Methodological*, 45(8):1232–1250, 2011.

[6] K. Cullinane and R. Bergqvist. Emission control areas and their impact on maritime transport. *Transportation Research Part D: Transport and Environment*, 28:1–5, 2014.

[7] CVX Research Inc. CVX: Matlab software for disciplined convex programming, version 2.0. `http://cvxr.com/cvx`, Aug. 2012.

[8] S. Dabia, E. Demir, and T. Van Woensel. An exact approach for the pollution-routing problem. *Transportation Science*, 2016.

[9] E. Demir, T. Bektaş, and G. Laporte. An adaptive large neighborhood search heuristic for the pollution-routing problem. *European Journal of Operational Research*, 223(2):346–359, 2012.

[10] Y. Du, Q. Chen, X. Quan, L. Long, and R. Y. Fung. Berth allocation considering fuel consumption and vessel emissions. *Transportation Research Part E: Logistics and Transportation Review*, 47(6):1021–1037, 2011.

[11] R. Eglese and T. Bektaş. Green vehicle routing. In P. Toth and D. Vigo, editors, *Vehicle Routing: Problems, Methods, and Applications*, pages 437–458. SIAM, 2014.

[12] K. Fagerholt, N. T. Gausel, J. G. Rakke, and H. N. Psaraftis. Maritime routing and speed optimization with emission control areas. *Transportation Research Part C: Emerging Technologies*, 52:57–73, 2015.

[13] K. Fagerholt, G. Laporte, and I. Norstad. Reducing fuel emissions by optimizing speed on shipping routes. *Journal of the Operational Research Society*, 61(3):523–529, 2010.

[14] K. Fagerholt and H. N. Psaraftis. On two speed optimization problems for ships that sail in and out of emission control areas. *Transportation Research Part D: Transport and Environment*, 39:56–64, 2015.

[15] R. Fukasawa, Q. He, and Y. Song. A disjunctive convex programming approach to the pollution-routing problem. *Transportation Research Part B: Methodological*, 94:61–79, 2016.

[16] R. A. Gatica and P. A. Miranda. A time based discretization approach for ship routing and scheduling with variable speed. *Networks and Spatial Economics*, 11(3):465–485, 2011.

[17] S. Gelareh and Q. Meng. A novel modeling approach for the fleet deployment problem within a short-term planning horizon. *Transportation Research Part E: Logistics and Transportation Review*, 46(1):76–89, 2010.

[18] M. Grant and S. Boyd. Graph implementations for nonsmooth convex programs. In V. Blondel, S. Boyd, and H. Kimura, editors, *Recent Advances in Learning and Control*, Lecture Notes in Control and Information Sciences, pages 95–110. Springer-Verlag Limited, 2008. `http://stanford.edu/~boyd/graph_dcp.html`.

[19] L. M. Hvattum, I. Norstad, K. Fagerholt, and G. Laporte. Analysis of an exact algorithm for the vessel speed optimization problem. *Networks*, 62(2):132–135, 2013.

[20] N. Katoh and T. Ibaraki. Resource allocation problems. *Handbook of combinatorial optimization*, 2:159–260, 1998.

[21] N. Katoh, A. Shioura, and T. Ibaraki. Resource allocation problems. In *Handbook of combinatorial optimization*, pages 2897–2988. Springer, 2013.

[22] C. Kontovas and H. N. Psaraftis. Reduction of emissions along the maritime intermodal container chain: operational models and policies. *Maritime Policy & Management*, 38(4):451–469, 2011.

[23] R. Kramer, N. Maculan, A. Subramanian, and T. Vidal. A speed and departure time optimization algorithm for the pollution-routing problem. *European Journal of Operational Research*, 247(3):782–787, 2015.

[24] R. Kramer, A. Subramanian, T. Vidal, and F. C. Lucídio dos Anjos. A matheuristic approach for the pollution-routing problem. *European Journal of Operational Research*, 243(2):523–539, 2015.

[25] Q. Meng and S. Wang. Optimal operating strategy for a long-haul liner service route. *European Journal of Operational Research*, 215(1):105–114, 2011.

[26] Q. Meng, S. Wang, H. Andersson, and K. Thun. Containership routing and scheduling in liner shipping: overview and future research directions. *Transportation Science*, 48(2):265–280, 2013.

[27] I. Norstad, K. Fagerholt, and G. Laporte. Tramp ship routing and scheduling with speed optimization. *Transportation Research Part C: Emerging Technologies*, 19(5):853–865, 2011.

[28] T. E. Notteboom. The time factor in liner shipping services. *Maritime Economics & Logistics*, 8(1):19–39, 2006.

[29] T. E. Notteboom and B. Vernimmen. The effect of high fuel costs on liner service configuration in container shipping. *Journal of Transport Geography*, 17(5):325–337, 2009.

[30] M. Patriksson. A survey on the continuous nonlinear resource allocation problem. *European Journal of Operational Research*, 185(1):1–46, 2008.

[31] H. N. Psaraftis and C. A. Kontovas. Speed models for energy-efficient maritime transportation: A taxonomy and survey. *Transportation Research Part C: Emerging Technologies*, 26:331–351, 2013.

[32] H. N. Psaraftis and C. A. Kontovas. Ship speed optimization: Concepts, models and combined speed-routing scenarios. *Transportation Research Part C: Emerging Technologies*, 44:52–69, 2014.

[33] H. N. Psaraftis and C. A. Kontovas. Slow steaming in maritime transportation: fundamentals, trade-offs, and decision models. In *Handbook of Ocean Container Transport Logistics*, pages 315–358. Springer, 2015.

[34] D. Ronen. The effect of oil price on the optimal speed of ships. *Journal of the Operational Research Society*, 33(11):1035–1040, 1982.

[35] D. Ronen. The effect of oil price on containership speed and fleet size. *Journal of the Operational Research Society*, 62(1):211–216, 2011.

[36] R. H. Tütüncü, K.-C. Toh, and M. J. Todd. Solving semidefinite-quadratic-linear programs using sdpt3. *Mathematical programming*, 95(2):189–217, 2003.

[37] United Nations Conference on Trade and Development. Review of maritime transportation 2016 - the long-term growth prospects for seaborne trade and maritime businesses, 2016.

[38] T. Vidal, D. Gribel, and P. Jaillet. Separable convex optimization with nested lower and upper constraints. *arXiv preprint arXiv:1703.01484*, 2017.

[39] T. Vidal, P. Jaillet, and N. Maculan. A decomposition algorithm for nested resource allocation problems. *SIAM Journal on Optimization*, 26(2):1322–1340, 2016.

[40] S. Wang and Q. Meng. Liner ship route schedule design with sea contingency time and port time uncertainty. *Transportation Research Part B: Methodological*, 46(5):615–633, 2012.

[41] S. Wang and Q. Meng. Sailing speed optimization for container ships in a liner shipping network. *Transportation Research Part E: Logistics and Transportation Review*, 48(3):701–714, 2012.

[42] S. Wang, Q. Meng, and Z. Liu. Bunker consumption optimization methods in shipping: A critical review and extensions. *Transportation Research Part E: Logistics and Transportation Review*, 53:49–62, 2013.

[43] S. Wang and X. Wang. A polynomial-time algorithm for sailing speed optimization with containership resource sharing. *Transportation Research Part B: Methodological*, 93:394–405, 2016.

[44] J. Xia, K. X. Li, H. Ma, and Z. Xu. Joint planning of fleet deployment, speed optimization, and cargo allocation for liner shipping. *Transportation Science*, 49(4):922–938, 2015.

[45] P. H. Zipkin. Simple ranking methods for allocation of one resource. *Management Science*, 26(1):34–43, 1980.

A Detailed computational results for instances in maritime and road transportation

Instance	Our algorithm		CVX/SDPT3	
	Objective	Time(s)	Objective	Time(s)
10-1	1111	9.00E-05	1111	0.94
10-2	854.69	2.74E-04	854.69	0.86
10-3	851.87	8.49E-05	851.87	0.80
10-4	659.06	7.23E-05	659.06	0.77
10-5	505.58	7.28E-05	505.58	0.89
10-6	943.97	8.35E-05	943.97	1.23
10-7	681.01	7.19E-05	681.01	0.77
10-8	849.11	1.10E-04	849.11	0.75
10-9	862.02	8.72E-05	862.02	0.81
10-10	628.8	7.37E-05	628.8	0.63
100-1	12553	2.74E-02	12553	1.50
100-2	13047	2.11E-02	13047	1.38
100-3	13183	4.13E-02	13183	1.30
100-4	12765	2.75E-02	12765	1.42
100-5	14041	4.21E-02	14041	1.14
100-6	14129	2.11E-02	14129	1.44
100-7	13250	2.04E-02	13250	1.45
100-8	11792	1.34E-02	11792	1.56
100-9	13317	2.12E-02	13317	1.92
100-10	12377	4.28E-02	12377	2.78
1000-1	159400	0.46	159400	9.23
1000-2	158970	0.23	158970	10.25
1000-3	155690	0.45	155690	10.19
1000-4	158910	0.47	158910	9.44
1000-5	154150	0.46	154150	9.45
1000-6	157770	0.44	157770	9.34
1000-7	158760	0.46	158760	9.92
1000-8	160020	0.45	160020	10.13
1000-9	158710	0.46	158710	9.56
1000-10	158160	0.48	158160	9.38

Table 7: Computational results of our algorithm and CVX/SDPT3 on the maritime instances with 10, 100, and 1000 nodes (The instance index n-k denotes the k-th instance of n nodes).

Instance	Our algorithm		CVX/SDPT3	
	Objective	Time(s)	Objective	Time(s)
10-1	0.5543	2.27E-04	0.5543	1.03
10-2	0.4220	3.76E-04	0.4220	0.81
10-3	0.3819	4.78E-04	0.3819	0.94
10-4	0.4855	2.43E-04	0.4855	0.80
10-5	0.3558	3.70E-04	0.3558	0.66
10-6	0.4485	2.44E-04	0.4485	0.67
10-7	0.3630	3.95E-04	0.3630	0.53
10-8	0.4509	3.89E-04	0.4509	0.59
10-9	0.5631	8.34E-04	0.5631	0.69
10-10	0.4889	7.35E-04	0.4889	0.64
100-1	5.028	8.13E-03	5.028	1.63
100-2	5.413	8.60E-03	5.413	1.75
100-3	5.684	7.16E-03	5.684	1.73
100-4	5.675	6.40E-03	5.675	1.59
100-5	5.453	6.92E-03	5.453	1.67
100-6	5.534	7.56E-03	5.534	1.73
100-7	5.331	5.94E-03	5.331	1.77
100-8	5.395	7.19E-03	5.395	1.70
100-9	5.270	6.84E-03	5.270	2.20
100-10	5.266	5.28E-03	5.266	1.98
1000-1	57.02	0.14	57.02	14.25
1000-2	55.33	0.13	55.33	15.13
1000-3	55.76	0.15	55.76	15.20
1000-4	56.17	0.14	56.17	15.41
1000-5	55.42	0.16	55.42	14.72
1000-6	54.02	0.16	54.02	15.02
1000-7	56.87	0.14	56.87	15.03
1000-8	56.31	0.14	56.31	15.14
1000-9	55.74	0.14	55.74	14.47
1000-10	55.47	0.14	55.47	15.19

Table 8: Computational results of our algorithm and CVX/SDPT3 on the road transportation instances with 10, 100, and 1000 nodes (The instance index n-k denotes the k-th instance of n nodes).