Energy-Efficient Timely Transportation of Long-Haul Heavy-Duty Trucks

Lei Deng
Dept. of IE, CUHK

Mohammad H. Hajiesmaili
Dept. of IE, CUHK

Minghua Chen
Dept. of IE, CUHK

Haibo Zeng
Dept. of ECE, Virginia Tech

ABSTRACT

We consider a timely transportation problem where a heavy-duty truck travels between two locations across the national highway system, subject to a hard deadline constraint. Our objective is to minimize the total fuel consumption of the truck, by optimizing both route planning and speed planning. The problem is important for cost-effective and environment-friendly truck operation, and it is uniquely challenging due to its combinatorial nature as well as the need of considering hard deadline constraint. We first show that the problem is NP-Complete; thus exact solution is computational prohibited unless P=NP. We then design a fully polynomial time approximation scheme (FPTAS) that attains an approximation ratio of $1+\epsilon$ with a network-size induced complexity of $O(mn^2/\epsilon^2)$, where m and n are the numbers of nodes and edges, respectively. While achieving highly-preferred theoretical performance guarantee, the proposed FPTAS still suffers from long running time when applying to national-wide highway systems with tens of thousands of nodes and edges. Leveraging elegant insights from studying the dual of the original problem, we design a fast heuristic solution with $O(m+n\log n)$ complexity. The proposed heuristic allows us to tackle the energy-efficient timely transportation problem on large-scale national highway systems. We further characterize a condition under which our heuristic generates an optimal solution. We observe that the condition holds in most of the practical instances in numerical experiments, justifying the superior empirical performance of our heuristic. We carry out extensive numerical experiments using real-world truck data over the actual U.S. highway network. The results show that our proposed solutions achieve 17% (resp. 14%) fuel consumption reduction, as compared to a fastest path (resp. shortest path) algorithm adapted from common practice.

CCS Concepts

•Applied computing → Transportation; •Mathematics of computing → *Mixed discrete-continuous optimization;*

e-Energy'16, June 21-24, 2016, Waterloo, ON, Canada
© 2016 ACM. ISBN 978-1-4503-4393-0/16/06…$15.00
DOI: http://dx.doi.org/10.1145/2934328.2934338

Keywords

Energy-efficient transportation; timely delivery; route planning; speed planning

1. INTRODUCTION

In the U.S., heavy-duty trucks haul more than 70% of all freight tonnage [11], and they consume 17.6% of energy in transportation sector [21, Tab. 2.8] and contribute to about 5% of the greenhouse gas emission [8]. Fuel cost is the largest operating cost (34%) of truck owners/operators [25], and reducing fuel consumption is critical for cost-effective and environment-friendly heavy-duty truck operations.

Currently there are mainly two lines of efforts to reduce fuel consumption of heavy-duty trucks. The first line is to operate with more fuel efficient trucks, from better designs for engines, drivetrains, aerodynamics, and tires [13,27,38], to better management of truck parts such as maintaining optimal tire pressures [4]. The second line is to operate heavy-duty trucks more economically. This explores several possibilities, e.g., reducing idling energy consumption [40], platooning more than one heavy-duty trucks [15,32], route planning [23,41,43], and speed planning [3,10,29,30]. In this paper, we focus on route and speed planning. Different routes could have different mileages, levels of congestion, road grades, and surface types, etc., all of which would largely affect the fuel consumption. Real-world studies [43] show that choosing a more efficient route for a heavy-duty truck can improve its fuel economy by 21%. Speed planning is another well recognized approach to effectively reduce fuel consumption: As a rule of thumb for truck operations on highway, every one mile per hour (mph) increase in speed incurs about 0.14 mile per gallon (mpg) penalty in fuel economy [3,10].

However, operating at low speed may result in excessive travel time and the goods carried by the truck cannot be delivered on time. We remark that timely delivery is critical for truck operators [12,35]. As estimated by the U.S. Federal Highway Administration (FHWA) in [35], unexpected delay can increase freight cost by 50% to 250%. Multiple reasons can explain the importance of timely delivery. First, some freight goods are perishable, such as food [18], which definitely require timely delivery. Second, to ensure customers' satisfaction, some companies, e.g., Amazon, may have a service-level agrement (SLA) with users, under which the delivery delay is guaranteed [6]. Finally, violating scheduled delay can introduce difficulties for global logistic decisions and even increase the uncertainty and inefficiency of supply chains [35]. Overall, it is crucial to ensure timely goods delivery for truck operators, and considering timely delivery in fuel cost minimization poses a unique challenge of which only partial results for special cases are recently available [29,30].

Motivated by the above observations, in this paper, we study the problem of *energy-efficient timely transportation* for heavy-duty trucks. We aim to minimize the heavy duty truck's fuel consumption while satisfying a hard deadline constraint, under which we take both route planning and speed planning into account to exploit complete design space of reducing fuel consumption. Since heavy-duty trucks are mainly operated for *long-haul* delivery and most of time run on highways [21, Tab. 5.2 and Fig. 5.1], we focus our model on their operation in the highway transportation network system. We summarize our contributions in the following.

▷ We formulate an energy-efficient timely transportation problem of minimizing the fuel consumption subject to a hard deadline constraint for a heavy-duty truck running on a highway transportation network, with design spaces of both route planning and speed planning in Sec. 2. We show that our problem is NP-Complete.

▷ In Sec. 3, we design a fully polynomial time approximation scheme (FPTAS) for solving the energy-efficient timely transportation problem. The proposed FPTAS attains an approximation ratio of $1+\epsilon$ with a network-size induced complexity of $O(mn^2/\epsilon^2)$, where m and n are the numbers of nodes and edges, respectively.

▷ While achieving highly-preferred theoretical performance guarantee, the proposed FPTAS still suffers from long running time when applying to national-wide highway systems with tens of thousands of nodes and edges. In Sec. 4, by leveraging elegant insights from studying the dual of the original problem, we design a fast heuristic solution with $O(m+n\log n)$ complexity. The proposed heuristic scheme allows us to tackle the energy-efficient timely transportation problem on large-scale national highway systems. We further characterize a condition under which our heuristic generates an optimal solution. We observe that the condition holds in most of the practical instances in numerical experiments in Sec. 5, justifying the superior empirical performance of our heuristic.

▷ We carry out extensive numerical experiments using real-world truck data over the U.S. highway network in Sec. 5. The results show that our proposed solutions achieve 17% (resp. 14%) fuel consumption reduction, as compared to a fastest path (resp. shortest path) algorithm adapted from common practice. The amount of fuel consumption saving is enough to power up more than 90% of the entire transportation sector in New York State [2].

▷ For those who are familiar with Restricted Shortest Path (RSP) problem [26,28,31], our energy-efficient timely transportation problem is a generalized version of RSP, including an extra design space of speed planning. Therefore, from the theoretical perspective, we generalize the FPTAS design and the dual-based design of RSP to our problem.

2. MODEL AND PROBLEM FORMULATION

2.1 System Model

Consider a highway transportation network as exemplified in Fig. 1. We model it as a directed graph $\mathcal{G}=(\mathcal{V},\mathcal{E})$, where $\mathcal{V}$ is the vertex/node set and $\mathcal{E}$ is the edge/road set. We define $n\triangleq|\mathcal{V}|$ as the number of nodes and $m\triangleq|\mathcal{E}|$ as the number of edges. For each edge $e\in\mathcal{E}$, we denote

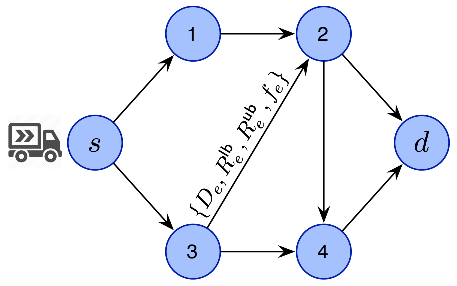


Figure 1: System model.

$D_e>0$ as its distance (unit: mile), and $R_e^{\mathsf{lb}}>0$ (resp. $R_e^{\mathsf{ub}}\geq R_e^{\mathsf{lb}}$) as its minimal (resp. maximal) speed (unit: mph). (Governments usually set the maximal speed for all highways and the minimal speed for some highways. For the sake of both safety and fuel efficiency, lower speed limits than passenger cars may be applied to large commercial vehicles like heavy-duty trucks and buses.) Now consider a long-haul heavy-duty truck who aims to ship cargos from a source node $s\in\mathcal{V}$ to a destination node $d\in\mathcal{V}$. The goal is to minimize the energy/fuel[1] consumption subject to a *hard* delay requirement $T>0$ (unit: hour).

Fuel consumption and travel delay are usually in conflict with each other, both of which are related to the speed profile of the truck. High travel speed obviously decreases the travel delay, but it can also increase the fuel consumption significantly [3, 10]. To analyze the performance tradeoff between energy and delay, we need to model the relationship between the fuel consumption and the travel speed. There are an intensive number of such models (see a survey in [22]). In this paper, we use the instantaneous fuel consumption model [14, 22] which generally depends on three factors: (i) static vehicle/road/environment properties, (ii) instantaneous acceleration/deceleration, and (iii) instantaneous speed. As we consider a specific vehicle running over a specific network, static vehicle/road/environment properties are fixed. The instantaneous acceleration/deceleration reflects the speed variation. However, since we consider a highway model, the truck spends most of time to maintain a relatively constant cruise speed [17, 36] such that the fuel consumption caused by acceleration/deceleration would be negligible. This motivates us to model the instantaneous fuel consumption as a function of the instantaneous speed.

We thus define $f_e:[R_e^{\mathsf{lb}},R_e^{\mathsf{ub}}]\to\mathbb{R}^+$ as the (instantaneous) *fuel-rate-speed function* of the truck running on edge e: if the vehicle's speed on edge e is r_e (unit: mph), the fuel consumption rate is $f_e(r_e)$ (unit: gallons per hour (gph)), and then the total fuel consumption for driving time τ (unit: hour) with the constant speed r_e is $f_e(r_e)\cdot\tau$ (unit: gallon). Since many existing models [14, 16, 17, 19, 39] use polynomial functions to model the fuel consumption which are also strictly convex in a reasonable speed limit region, in this paper, we assume that $f_e(\cdot)$ is a polynomial function and is strictly convex[2] over $[R_e^{\mathsf{lb}},R_e^{\mathsf{ub}}]$. This assumption also holds in the physical interpretation of fuel-rate-speed function as shown in our technical report [24], and is further verified in our simulation using real-world data (see Fig. 5(a)).

[1]We interchangeably use *fuel* and *energy* in this paper.
[2]The strict convexity can be relaxed to convexity. For simplicity, we use the strict convexity in this paper.

2.2 Problem Formulation

We consider two design spaces: path selection (route planning) and speed optimization (speed planning). For path selection, we define a binary variable x_e for any $e \in \mathcal{E}$,

$$x_e = \begin{cases} 1, & \text{Edge } e \text{ is on the selected path;} \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

For the speed optimization, the truck needs to determine a speed profile (speeds at all travel time) over any selected edge. This is a functional variable, but the convexity of fuel-rate-speed function can simplify the speed profile significantly based on the following lemma.

Lemma 1. *For any edge e, if the travel time t_e is given, i.e., the truck must pass edge e with exactly t_e hours, then the optimal speed profile to minimize the fuel consumption is to maintain constant speed D_e/t_e during the whole trip.*

Lemma 1 shows that for any edge, any non-constant speed profile is dominated by another constant speed profile in terms of fuel consumption without sacrificing the delay performance. Therefore, without loss of optimality, the truck only needs to follow a constant speed for any edge. As explained in Sec. 2.1, since we consider a long-haul highway scenario, we will ignore the speed transition period between two adjacent edges. Thus, for the speed optimization, we consider the travel time $t_e > 0$ over each edge e as the design variable, which equivalently implies a constant speed D_e/t_e over e. We then define a *fuel-time* function $c_e(\cdot)$ for each road e,

$$c_e(t_e) \triangleq t_e \cdot f_e(\frac{D_e}{t_e}), \tag{2}$$

which is the total fuel consumption for the truck traveling edge e with travel time t_e.

By vectorizing our decision variables as $\boldsymbol{x} \triangleq \{x_e : e \in \mathcal{E}\}$ and $\boldsymbol{t} \triangleq \{t_e : e \in \mathcal{E}\}$, now we are ready to formulate our PAth selection and Speed Optimization (PASO) problem,

$$\mathsf{PASO}: \min_{\boldsymbol{x}\in\mathcal{X},\boldsymbol{t}\in\mathcal{T}} \quad \sum_{e\in\mathcal{E}} x_e \cdot c_e(t_e) \tag{3}$$

$$\text{s.t.} \quad \sum_{e\in\mathcal{E}} x_e t_e \le T, \tag{4}$$

In PASO, set $\mathcal{X}$ restricts that one and only one $s-d$ path is selected, defined as

$$\mathcal{X} \triangleq \{\boldsymbol{x} : x_e \in \{0,1\}, \forall e \in \mathcal{E}, \text{ and } \sum_{e\in\mathsf{out}(v)} x_e - \sum_{e\in\mathsf{in}(v)} x_e = 1_{\{v=s\}} - 1_{\{v=d\}}, \forall v \in \mathcal{V}\},$$

where $1_{\{\cdot\}}$ is the indicator function, $\mathsf{in}(v) \triangleq \{(u,v) : (u,v) \in \mathcal{E}\}$ is the set of incoming edges of node $v \in \mathcal{V}$, $\mathsf{out}(v) \triangleq \{(v,u) : (v,u) \in \mathcal{E}\}$ is the set of outgoing edges of node v. Set $\mathcal{T}$ captures the speed limits of all roads, defined as

$$\mathcal{T} \triangleq \{\boldsymbol{t} : t_e^{\mathsf{lb}} \le t_e \le t_e^{\mathsf{ub}}, \forall e \in \mathcal{E}\},$$

where $t_e^{\mathsf{lb}} \triangleq \frac{D_e}{R_e^{\mathsf{ub}}}$ and $t_e^{\mathsf{ub}} \triangleq \frac{D_e}{R_e^{\mathsf{lb}}}$ are the minimal and maximal travel time due to the speed limits on edge e, respectively. Constraint (4) is to satisfy the hard delay requirement. Objective (3) is to minimize the total fuel consumption over the selected path.

2.3 Complexity Hardness

PASO has both integer variables and continuous variables. Thus it is worth understanding its hardness first. It turns out that a special case of PASO is the well-known Restricted Shortest Path (RSP) problem [26, 28]. In RSP, a directed graph is given where each edge has a *fixed* travel time and travel cost, and the goal is to find a minimal-cost path subject to a hard path delay requirement. Clearly, our problem PASO generalizes RSP where we allow a varying edge cost and edge time because of the design space of speed optimization. Since RSP is NP-Complete [26], we can thus easily prove that our problem PASO is also NP-Complete.

Theorem 1. *PASO is NP-Complete.*

Proof. We can prove it by setting $R_e^{\mathsf{lb}} = R_e^{\mathsf{ub}}$ to an appropriate value for each edge e in PASO, and using the result that RSP is NP-Complete [26]. $\square$

2.4 Preprocessing and Some Notations

We first check the feasibility of our problem PASO. We can use the shortest path algorithm where each edge e has cost t_e^{lb} to find the fastest path. If the travel time of the fastest path is larger than the delay requirement T, PASO is infeasible. In the rest of this paper, we thus assume that the delay constraint T is at least the travel time of the fastest path such that the problem is feasible.

We then analyze properties of the fuel-time function $c_e(\cdot)$.

Lemma 2. *$c_e(t_e)$ is strictly convex over $[t_e^{\mathsf{lb}}, t_e^{\mathsf{ub}}]$. Also, there exists a point $\hat{t}_e \in [t_e^{\mathsf{lb}}, t_e^{\mathsf{ub}}]$ such that $c_e(t_e)$ is first strictly decreasing over $[t_e^{\mathsf{lb}}, \hat{t}_e]$ and then strictly increasing over $[\hat{t}_e, t_e^{\mathsf{ub}}]$.*

Based on Lemma 2, we can easily prove that the travel time over edge e, i.e., t_e, in any optimal solution of PASO must be in the region $[t_e^{\mathsf{lb}}, \hat{t}_e]$. Otherwise, we can decrease the trave time from t_e to $\hat{t}_e$ and at the same time decrease the fuel consumption, which violates the optimality of t_e. Thus, without loss of optimality, we can reset the travel time limit from $[t_e^{\mathsf{lb}}, t_e^{\mathsf{ub}}]$ to $[t_e^{\mathsf{lb}}, \hat{t}_e]$, which equivalently implies that we reset the speed limit from $[R_e^{\mathsf{lb}}, R_e^{\mathsf{ub}}]$ to $[D_e/\hat{t}_e, R_e^{\mathsf{ub}}]$. After such preprocessing, in the rest of the paper, $c_e(t_e)$ can be assumed to be *strictly convex and strictly decreasing* over $t_e \in [t_e^{\mathsf{lb}}, t_e^{\mathsf{ub}}]$ *without loss of optimality.*

In the rest of the paper, define an $s-d$ path p as the set of all *edges* over p and $\boldsymbol{t}_p \triangleq \{t_e : e \in p\}$ as the corresponding travel time set. Moreover, we define $c(p, \boldsymbol{t}_p) \triangleq \sum_{e\in p} c_e(t_e)$ as the fuel consumption of path p with travel time set $\boldsymbol{t}_p$, and OPT as the optimal value of PASO.

Next, we will propose a fully polynomial time approximation scheme (FPTAS) in Sec. 3 and a fast dual-based heuristic scheme in Sec. 4 to solve our problem PASO.

3. AN FPTAS FOR PASO

Since PASO generalizes RSP, which is well-known to have an FPTAS [28, 34], it is natural to ask whether we can extend RSP's FPTAS for our problem PASO. In this section, by carefully tackling the difference between PASO and RSP, we "reformulate" PASO such that we can adapt RSP's FPTAS to construct an FPTAS for PASO. More specifically, in this section, we propose an approximation scheme (Algorithm 3) such that for any given $\epsilon \in (0,1)$, it can find a

$(1+\epsilon)$-approximate solution in the sense that the solution is *feasible* and the corresponding fuel consumption is at most $(1+\epsilon)\mathsf{OPT}$, and the time complexity is polynomial in both the problem size and $\frac{1}{\epsilon}$.

The essence of RSP's FPTAS [28, 34] is a test procedure. For any input value $V > 0$ and any input accuracy parameter $\delta > 0$, the test procedure can "approximately" compare V and the optimal value OPT in the sense that it can tell either $\mathsf{OPT} > V$ or $\mathsf{OPT} \leq (1+\delta)V$ in polynomial time. Based on this test procedure, starting with some arbitrary lower bound LB and upper bound UB for OPT, a binary search scheme is designed [28, 34] to exponentially narrow down the bounding interval $[\mathsf{LB}, \mathsf{UB}]$ and finally a $(1+\epsilon)$-approximate solution is outputted.

To solve our problem PASO, we adapt RSP's FPTAS by designing our own test procedure. In RSP, [28] and [34] use the *rounding and scaling* technique, where each *fixed* edge cost is rounded into certain (polynomial) number of cost levels controlled by the accuracy parameter δ. As we only require an "approximate" comparison, rounding into certain number of cost levels is enough to perform such a task. However, as opposed to a fixed edge cost in RSP, in PASO each edge has a fuel-time function. Hence, instead of rounding a fixed cost in RSP, we *quantize* the continuous fuel-time function $c_e(\cdot)$ into another staircase fuel-time function $\tilde{c}_e(\cdot)$ according to the input value V and the input accuracy parameter δ, which can be further characterized by a polynomial number of *representative points*. We then prove that such quantization can perform the "approximate" comparison.

Later on we will describe our algorithms in a bottom-up fashion. We first describe the quantizing procedure (Algorithm 1) in Sec. 3.1. Then we present our own test procedure (Algorithm 2) which invokes Algorithm 1 in Sec. 3.2. Finally, we describe the whole FPTAS (Algorithm 3) which invokes Algorithm 2 in Sec. 3.3.

3.1 Quantizing Fuel-Time Function

For any input value $V > 0$ and $N \in \mathbb{Z}^+$, we quantize the edge-e fuel-time function $c_e(t_e)$ to be

$$\tilde{c}_e(t_e) \triangleq \min\left\{\left\lfloor \frac{c_e(t_e)}{V} \right\rfloor + 1, N\right\}, \forall t_e \in [t_e^{\mathsf{lb}}, t_e^{\mathsf{ub}}]. \quad (5)$$

Since we have assumed that $c_e(t_e)$ is strictly decreasing in Sec. 2.4, $\tilde{c}_e(t_e)$ thus becomes a *staircase* function with at most N stairs. During the quantization, parameter V is to control the accuracy, which is the vertical span of each stair. Larger V means rougher quantization and lower accuracy but smaller complexity. Parameter N is to control the maximal number of stairs. Since $c_e(t_e)$ could take an arbitrarily large value, the number of stairs could be unbounded, which definitely incurs high complexity. To design a polynomial time test procedure where we only need to perform an "approximate" comparison, we truncate $c_e(t_e)$ by putting a ceil VN. This truncation is sufficient for use in the test procedure (see Sec. 3.2). Clearly, $\tilde{c}_e(t_e)$ is a *quantized and truncated* version of $c_e(t_e)$. An example is shown in Fig. 2. Here we set $V = 20, N = 4$. Thus, each stair spans 20 and $c_e(t_e)$ is truncated by the ceil $VN = 80$. The resulting curve $\tilde{c}_e(t_e)$ is a non-increasing staircase function, which jumps from 4 to 3 at $t_e = 1.8$ and jumps from 3 to 2 at $t_e = 2.8$.

Moreover, since $\tilde{c}_e(t_e)$ is a staircase function and only

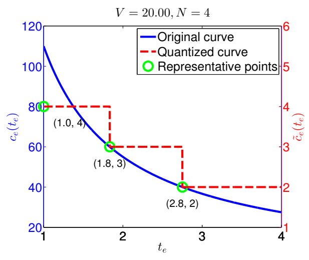


Figure 2: An example for quantizing $c_e(\cdot)$.

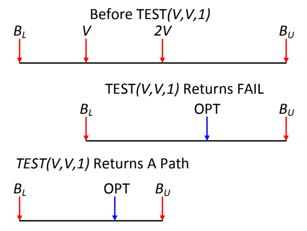


Figure 3: Binary search (*Step 2*) of Algorithm 3.

Algorithm 1 A Quantizing Procedure $\texttt{QUANTIZE}(e, V, N)$

```
1: for $i = 1, 2, \cdots, N$ do
2:     Set $\tau_e^i = \mathsf{nan}$
3: end for
4: Set $n_{\min} = \tilde{c}_e(t_e^{\mathsf{ub}}) = \min\{\lfloor \frac{c_e(t_e^{\mathsf{ub}})}{V} \rfloor + 1, N\}$
5: Set $n_{\max} = \tilde{c}_e(t_e^{\mathsf{lb}}) = \min\{\lfloor \frac{c_e(t_e^{\mathsf{lb}})}{V} \rfloor + 1, N\}$
6: Set $\tau_e^{n_{\max}} = t_e^{\mathsf{lb}}$
7: for $i = n_{\min}, n_{\min}+1, \cdots, n_{\max}-1$ do
8:     Solve the equation $c_e(t_e) = iV$ over $t_e \in [t_e^{\mathsf{lb}}, t_e^{\mathsf{ub}}]$
9:     if the equation has a solution $t_e$ then
10:        Set $\tau_e^i = t_e$
11:    end if
12: end for
13: return $\boldsymbol{\tau}_e = (\tau_e^1, \tau_e^2, \cdots, \tau_e^N)$
```

takes integer values, we can use an N-dim vector $\boldsymbol{\tau}_e$ to represent it without any information loss. We define it as $\boldsymbol{\tau}_e \triangleq (\tau_e^1, \tau_e^2, \cdots, \tau_e^N)$ where τ_e^i is the minimal travel time over $[t_e^{\mathsf{lb}}, t_e^{\mathsf{ub}}]$ such that $\tilde{c}_e(\cdot) = i$ and is defined as nan if $\tilde{c}_e(\cdot) = i$ has no solution. For the example in Fig. 2, we have $\tau_e = (\tau_e^1, \tau_e^2, \tau_e^3, \tau_e^4) = (\mathsf{nan}, 2.8, 1.8, 1)$.

We call (τ_e^i, i) the i-th *representative point* of $\tilde{c}_e(\cdot)$. Thus $\tilde{c}_e(\cdot)$ is characterized by at most N representative points, which will play a key role in our test procedure in Sec. 3.2. We summary the quantizing procedure $\texttt{QUANTIZE}(e, V, N)$ in Algorithm 1. The basic idea is to first find the range of the stair levels, i.e., $[n_{\min}, n_{\max}]$ and then find τ_e^i for any level i in this range by solving an equation $c_e(t_e) = iV$.

Time Complexity: The major complexity of Algorithm 1 comes from line 8, which needs to solve an equation. Since we have assumed that $c_e(t_e)$ is a strictly decreasing function, we can use a binary search to solve this equation, which has time complexity $O(\log(t_e^{\mathsf{ub}} - t_e^{\mathsf{lb}}))$[3]. Hence, the total complexity of $\texttt{QUANTIZE}(e, V, N)$ is $O(N \log(t_e^{\mathsf{ub}} - t_e^{\mathsf{lb}}))$. If we define

$$\xi \triangleq \max_{e \in \mathcal{E}} \left(t_e^{\mathsf{ub}} - t_e^{\mathsf{lb}}\right) \quad (6)$$

as the maximal range of travel time over all edges, for any $e \in \mathcal{E}$, the complexity of $\texttt{QUANTIZE}(e, V, N)$ is $O(N \log \xi)$.

3.2 The Test Procedure

[3]We normally cannot solve an equation exactly, but we should ensure some precision/tolerance level. Precisely, the complexity should be $O(\log\left(\frac{t_e^{\mathsf{ub}} - t_e^{\mathsf{lb}}}{\mathsf{tol}}\right))$ where tol is the tolerance level. For simplicity, we do not discuss this precision/tolerance issue in this paper.

As introduced above, the test procedure should "approximately" compare V and the optimal value OPT such that it can answer either $\mathsf{OPT} > V$ or $\mathsf{OPT} \le (1+\delta)V$ in polynomial time. Inspired by [34], which improves the FPTAS of RSP in [28], we adopt a more powerful test procedure, denoted by $\texttt{TEST}(L, U, \delta)$. It can answer either $\mathsf{OPT} > U$ or $\mathsf{OPT} \le U + \delta L$. Clearly, if we set $L = U = V$, $\texttt{TEST}(V, V, \delta)$ can answer either $\mathsf{OPT} > V$ or $\mathsf{OPT} \le (1+\delta)V$, which exactly completes the "approximate" comparison. The reason to adopt a more powerful test procedure, similar to [34], is that we will also use it to finally output a $(1+\epsilon)$-approximate solution. We will discuss it soon in Sec. 3.3.

The details of $\texttt{TEST}(L, U, \delta)$ are shown in Algorithm 2. As we mentioned before, the major difference between our problem PASO and the existing problem RSP is that PASO has a continuous fuel-time function for each edge instead of a fixed cost. Thus, different from the test procedure for RSP (see [34, Fig. 1]), we have a step to invoke the quantizing procedure (Algorithm 1) to quantize the fuel-time function, as shown in lines 3-5 in Algorithm 2. More importantly, since our test procedure $\texttt{TEST}(L, U, \delta)$ aims to check either $\mathsf{OPT} > U$ or $\mathsf{OPT} \le U + \delta L$, roughly speaking, we do not need to quantize the portion of each fuel-time function with high fuel cost, i.e., larger than $U + \delta L$. Hence, to ensure polynomial time complexity eventually, we put a ceil $V(N+1)$ for $c_e(t_e)$ as shown in line 4 of the algorithm, where V and N are appropriately set such that $V(N+1) \ge U + \delta L$.

After such quantization, the fuel-time function $c_e(t_e)$ for each edge e consists of at most $N+1$ representative points. Therefore, conceptually we can construct a new graph $\tilde{\mathcal{G}} = (\mathcal{V}, \tilde{\mathcal{E}})$. Each edge $e \in \mathcal{E}$ in the original graph corresponds to at most $N+1$ edges in the new graph $\tilde{\mathcal{E}}$. For each edge $e \in \tilde{\mathcal{E}}$, the edge cost $\tilde{c}_e$ is a positive integer, as shown in (5). This is exactly an RSP problem. Therefore, the remaining steps follow the test procedure for RSP on the new graph $\tilde{\mathcal{G}}$. Specifically, since each edge $e \in \mathcal{E}$ has at most $N+1$ possible cost values all of which are positive integers (each edge e in the new graph $\tilde{\mathcal{E}}$ has a positive integer cost), we can use dynamic programming to complete such test. Similar to [28, 34], we define $g_v(c)$ as the minimal path travel time among all $s-v$ paths whose path cost is at most $c \in \mathbb{Z}^+$, and define $g_v(c) = \infty$ if no such path. The optimality condition (or Bellman's equation) becomes, for any $c = 1, 2, \cdots$,

$$g_v(c) = \min\{g_v(c-1), \min_{u,i:e=(u,v)\in\mathcal{E}, i=1,\cdots,N, \tau_e^i \neq \mathsf{nan}} \{g_u(c-i) + \tau_e^i\}\} \quad (7)$$

which is shown in line 10 in Algorithm 2. Since we only need to answer either $\mathsf{OPT} > U$ or $\mathsf{OPT} \le U + \delta L$, we do not have to process large c. Instead, iterating c from 1 to N is enough for us to complete this task. This dynamic programming procedure is shown in lines 6-15 of Algorithm 2.

In PASO, we should carefully design the quantizing and the dynamic programming procedures jointly to guarantee performance, as shown in the following lemmas, which are the counterparts to Lemma 2 and Lemma 3 for RSP in [34].

Lemma 3. *If Algorithm 2 returns a path p and travel time set $\boldsymbol{t}_p$, then we have*

$$\mathit{OPT} \le c(p, \boldsymbol{t}_p) \le U + L\delta. \quad (8)$$

Algorithm 2 A Test Procedure $\texttt{TEST}(L, U, \delta)$

```
 1: Set $V = \frac{L\delta}{n+1}$
 2: Set $N = \lfloor \frac{U}{V} \rfloor + n + 1$
 3: **for** $e \in \mathcal{E}$ **do**
 4:     Get $\boldsymbol{\tau}_e = \texttt{QUANTIZE}(e, V, N+1)$
 5: **end for**
 6: Set $g_s(c) = 0, \forall c = 0, 1, \cdots, N$
 7: Set $g_v(0) = \infty, \forall v \neq s, v \in \mathcal{V}$
 8: **for** $c = 1, 2, \cdots, N$ **do**
 9:     **for** $v \in \mathcal{V}$ **do**
10:         Set $g_v(c)$ according to (7)
11:     **end for**
12:     **if** $g_d(c) \le T$ **then**
13:         **return** the corresponding path $p$ and travel time
            set $\boldsymbol{t}_p = \{t_e : e \in p\}$
14:     **end if**
15: **end for**
16: **return** FAIL
```

Lemma 4. *If $U \ge \mathit{OPT}$, then Algorithm 2 must return a feasible path p and travel time set $\boldsymbol{t}_p$, which satisfy*

$$c(p, \boldsymbol{t}_p) \le \mathit{OPT} + L\delta. \quad (9)$$

Lemma 5. *If Algorithm 2 returns FAIL, then we have*

$$\mathit{OPT} > U. \quad (10)$$

PROOF. This directly follows Lemma 4. □

Our test procedure either returns a path p and travel time set $\boldsymbol{t}_p$ in line 13, which implies that $\mathsf{OPT} \le U + L\delta$ from Lemma 3, or returns FAIL in line 16, which implies $\mathsf{OPT} > U$ from Lemma 5. Therefore, Lemma 3 and Lemma 5 justify that our test procedure (Algorithm 2) completes the "approximate" comparison, i.e., answers either $\mathsf{OPT} > U$ or $\mathsf{OPT} \le U + L\delta$.

Thus, for the purpose of the test procedure, Lemma 3 and Lemma 5 are enough. However, we present Lemma 4, which is stronger than Lemma 5, to provide a sufficient condition such that our test procedure returns a path p and travel time set $\boldsymbol{t}_p$. We will use Lemma 4 shortly in Sec. 3.3 to finally output a $(1+\epsilon)$-approximate solution.

Time Complexity: The quantizing procedures for all edges in lines 3-5 require $O(mN\log\xi)$. The dynamic programming procedure in lines 6-15 requires $O(mN^2)$. Since $N = \lfloor \frac{U}{V} \rfloor + n + 1 = \lfloor \frac{U}{L} \cdot \frac{n+1}{\delta} \rfloor + n + 1 = O(\frac{U}{L} \cdot \frac{n}{\delta} + n)$, the total time complexity of Algorithm 2 is $O(mN\log\xi + mN^2) = O(m(\frac{U}{L} \cdot \frac{n}{\delta} + n)\log\xi + m(\frac{U}{L} \cdot \frac{n}{\delta} + n)^2)$.

3.3 The Proposed FPTAS

Based on our own test procedure (Algorithm 2), we then follow the FPTAS for RSP in [34, Fig. 2] by replacing its test procedure with ours. For completeness, we present the FPTAS in Algorithm 3 and explain it with the following three steps.

Step 1 (line 1): To initialize the bound interval, we need to first obtain a lower bound LB and an upper bound UB for the optimal value OPT. Define that the minimal single-edge fuel cost is $\mathsf{C}^{\mathsf{lb}} \triangleq \min_{e\in\mathcal{E}} c_e(t_e^{\mathsf{ub}})$ and the maximal single-edge fuel cost is $\mathsf{C}^{\mathsf{ub}} \triangleq \max_{e\in\mathcal{E}} c_e(t_e^{\mathsf{lb}})$. Simply, we can use the minimal single-edge fuel consumption C^{lb} as the lower bound LB and use the maximal single-path[4] fuel consumption $n\mathsf{C}^{\mathsf{ub}}$ as the

[4]A simple path can have at most n edges.

Algorithm 3 An FPTAS

```
1: Get a lower bound LB and upper bound UB for OPT
2: Set $B_L = \mathsf{LB}$
3: Set $B_U = \mathsf{UB}$
4: while $\frac{B_U}{B_L} > 16$ do
5:     $V = \sqrt{B_L \cdot B_U}$
6:     Call TEST$(V, V, 1)$
7:     if TEST$(V, V, 1)$ returns FAIL then
8:         Set $B_L = V$
9:     else
10:        Set $B_U = 2V$
11:    end if
12: end while
13: Call TEST$(B_L, B_U, \epsilon)$
```

upper bound UB. Also, in Sec. 4, we will propose a heuristic scheme which can always output a set of LB and UB.

Step 2 (lines 2-12): Using the initial lower bound LB and upper bound UB, we design a binary search scheme, which repeatedly invokes our test procedure (Algorithm 2) to exponentially narrow down the bound interval $[B_L, B_U]$ until $B_U/B_L \le 16$. The binary search step is visualized in Fig. 3. Note that we always keep B_L as a lower bound and B_U as an upper bound for OPT. Whenever $B_U/B_L > 16$, we input the geometric mean $V = \sqrt{B_L \cdot B_U}$ and $\delta = 1$ to the test procedure, as shown in lines 5 and 6. If TEST$(V, V, 1)$ returns FAIL, then according to Lemma 4, we must have $V < \mathsf{OPT}$. In this case, we reset the lower bound B_L to be V in line 8. Otherwise, TEST$(V, V, 1)$ returns a feasible path p and travel time set $\boldsymbol{t}_p$. According to Lemma 3, we must have $\mathsf{OPT} \le V + \delta V = 2V$. We reset the upper bound to be $2V$ in line 10. It can be easily shown that this binary search returns a lower bound B_L and an upper bound B_U for OPT such that $B_U/B_L \le 16$ in $O(\log\log\frac{\mathsf{UB}}{\mathsf{LB}})$ iterations.

Step 3 (line 13): When $\frac{B_U}{B_L} \le 16$, we call our test procedure again but we use $L = B_L$ and $U = B_U$ and $\delta = \epsilon$. Since $B_U \ge \mathsf{OPT}$, according to Lemma 4, TEST(B_L, B_U, ϵ) must return a feasible path p and travel time $\boldsymbol{t}_p$ such that

$$c(p, \boldsymbol{t}_p) \le \mathsf{OPT} + \epsilon B_L \le \mathsf{OPT} + \epsilon\mathsf{OPT} = (1+\epsilon)\mathsf{OPT}.$$

Therefore, we get a $(1+\epsilon)$-approximate solution to PASO.

Time Complexity: *Step 1* requires $O(m)$ to get an initial lower bound LB and upper bound UB. *Step 2* invokes the test procedure $O(\log\log\frac{\mathsf{UB}}{\mathsf{LB}})$ times and each invoke takes $O(mn\log\xi + mn^2)$ time by using $L = U = V$ and $\delta = 1$. Thus *Step 2* takes $O((mn\log\xi + mn^2)\log\log\frac{\mathsf{UB}}{\mathsf{LB}})$. *Step 3* also invokes the test procedure, and it takes $O(\frac{mn\log\xi}{\epsilon} + \frac{mn^2}{\epsilon^2})$ time by using $\delta = \epsilon < 1$ and $O(\frac{U}{L}) = O(\frac{B_U}{B_L}) = O(1)$ because $\frac{B_U}{B_L} \le 16 = O(1)$. Here we can also see why we need to use a binary search to obtain $\frac{B_U}{B_L} \le 16$ in *Step 2*. This is because $\frac{B_U}{B_L} = O(1)$ ensures polynomial time complexity in *Step 3*. Therefore, the total complexity is $O((mn\log\xi + mn^2)\log\log\frac{\mathsf{UB}}{\mathsf{LB}} + \frac{mn\log\xi}{\epsilon} + \frac{mn^2}{\epsilon^2})$.

We summarize our results for the approximate scheme in the following theorem.

Theorem 2. *Algorithm 3 returns a $(1+\epsilon)$-approximate solution for PASO in time $O((mn\log\xi + mn^2)\log\log\frac{UB}{LB} + \frac{mn\log\xi}{\epsilon} + \frac{mn^2}{\epsilon^2})$. In addition, when we use $LB = \mathsf{C}^{\mathsf{lb}}$ and $UB = n\mathsf{C}^{\mathsf{ub}}$ where $\mathsf{C}^{\mathsf{lb}} \triangleq \min_{e\in\mathcal{E}} c_e(t_e^{\mathsf{ub}})$ and $\mathsf{C}^{\mathsf{ub}} \triangleq \max_{e\in\mathcal{E}} c_e(t_e^{\mathsf{lb}}) = c_{e_1}(t_{e_1}^{\mathsf{lb}})$, we have $\log\log\frac{UB}{LB} = \max\{O(\log\log n), O(I_{e_1})\}$ where I_{e_1} is the input size of all parameters of edge e_1. Thus, Algorithm 3 has time complexity polynomial in the input size of the problem PASO and therefore is an FPTAS.*

Although we generalize the FPTAS design from RSP to PASO, such an FPTAS (Algorithm 3) still has high complexity for a large-scale highway network with tens of thousands of nodes and edges. In the next section, we propose a heuristic scheme with substantially lower complexity.

4. A FAST DUAL-BASED HEURISTIC

In this section, we present a heuristic scheme for our problem PASO based on Lagrangian relaxation. Such a heuristic scheme, as we will show later in Sec. 4.3, runs much faster than the FPTAS (Algorithm 3). Also, it always outputs a lower bound LB and an upper bound UB on OPT, which implements *Step 1* in Algorithm 3. Moreover, in most practical scenarios as shown in Sec. 5, this heuristic scheme outputs an optimal (or at least near optimal) solution, i.e., $\mathsf{LB} = \mathsf{UB} = \mathsf{OPT}$ (or at least $\mathsf{LB} \approx \mathsf{OPT} \approx \mathsf{UB}$).

4.1 Lagrangian Relaxation and Dual Problem

In our problem PASO, since the hard delay constraint (4) couples path selection variable $\boldsymbol{x}$ with speed optimization variable $\boldsymbol{t}$, we relax it and introduce a Lagrangian dual variable $\lambda \ge 0$, which can be interpreted as a (per-unit) delay price *over the entire network*.

Based on such relaxation, we can get the corresponding Lagrangian,

$$\begin{aligned} L(\boldsymbol{x}, \boldsymbol{t}, \lambda) &\triangleq \sum_{e\in\mathcal{E}} x_e \cdot c_e(t_e) + \lambda(\sum_{e\in\mathcal{E}} x_e t_e - T) \\ &= \sum_{e\in\mathcal{E}} x_e \cdot (c_e(t_e) + \lambda t_e) - \lambda T, \qquad (11) \end{aligned}$$

and the corresponding dual function is defined as $D(\lambda) \triangleq \min_{\boldsymbol{x}\in\mathcal{X}, \boldsymbol{t}\in\mathcal{T}} L(\boldsymbol{x}, \boldsymbol{t}, \lambda)$. Then the dual problem of PASO is formulated as

$$(\mathsf{PASO\text{-}Dual}) \qquad \max_{\lambda\ge 0} D(\lambda)$$

4.2 Obtain Dual Function

Before we solve the dual problem, let us first show how to obtain the dual function for a given λ as follows,

$$\begin{aligned} D(\lambda) &= \min_{\boldsymbol{x}\in\mathcal{X}, \boldsymbol{t}\in\mathcal{T}} L(\boldsymbol{x}, \boldsymbol{t}, \lambda) \\ &= -\lambda T + \min_{\boldsymbol{x}\in\mathcal{X}, \boldsymbol{t}\in\mathcal{T}} \sum_{e\in\mathcal{E}} x_e \cdot (c_e(t_e) + \lambda t_e) \\ &\overset{(E_1)}{=} -\lambda T + \min_{\boldsymbol{x}\in\mathcal{X}} \left[\min_{\boldsymbol{t}\in\mathcal{T}} \sum_{e\in\mathcal{E}} x_e \cdot (c_e(t_e) + \lambda t_e) \right] \\ &\overset{(E_2)}{=} -\lambda T + \min_{\boldsymbol{x}\in\mathcal{X}} \sum_{e\in\mathcal{E}} x_e \cdot \min_{t_e^{\mathsf{lb}} \le t_e \le t_e^{\mathsf{ub}}} (c_e(t_e) + \lambda t_e) \\ &\overset{(E_3)}{=} -\lambda T + \min_{\boldsymbol{x}\in\mathcal{X}} \sum_{e\in\mathcal{E}} x_e \cdot [c_e(t_e^*(\lambda)) + \lambda t_e^*(\lambda)] \\ &\overset{(E_4)}{=} -\lambda T + \min_{\boldsymbol{x}\in\mathcal{X}} \sum_{e\in\mathcal{E}} x_e \cdot w_e(\lambda) \\ &\overset{(E_5)}{=} -\lambda T + \sum_{e\in p^*(\lambda)} w_e(\lambda). \qquad (12) \end{aligned}$$

We explain $(E_1)-(E_5)$ in (12) one by one. Equality (E_1) is because no coupled constraints exist for $\boldsymbol{x}$ and $\boldsymbol{t}$. Equality (E_2) is because no coupled constraints exist for the travel time at different edges in $\mathcal{T}$.

In equality (E_3), $t_e^*(\lambda)$ is defined as

$$t_e^*(\lambda) \triangleq \arg \min_{t_e^{\mathsf{lb}} \le t_e \le t_e^{\mathsf{ub}}} \left(c_e(t_e) + \lambda t_e\right). \tag{13}$$

Note that since we have assumed that $c_e(t_e)$ is strictly convex and strictly decreasing over $[t_e^{\mathsf{lb}}, t_e^{\mathsf{ub}}]$ in Sec. 2.4, $t_e^*(\lambda)$ is unique and thus (13) is well defined. Specifically, $t_e^*(\lambda)$ can be obtained analytically as follows.

Lemma 6. *Define $c_e'^{-1}(\cdot)$ as the inverse function of $c_e(\cdot)$. Then we have*

$$t_e^*(\lambda) = \begin{cases} t_e^{\mathsf{ub}}, & \text{If } 0 \le \lambda < -c_e'(t_e^{\mathsf{ub}}); \\ c_e'^{-1}(-\lambda), & \text{If } -c_e'(t_e^{\mathsf{ub}}) \le \lambda \le -c_e'(t_e^{\mathsf{lb}}); \\ t_e^{\mathsf{lb}}, & \text{If } \lambda > -c_e'(t_e^{\mathsf{lb}}). \end{cases} \tag{14}$$

In addition, (13) has a nice economic interpretation. As we have relaxed the hard delay constraint, we penalize each edge e with a delay cost, which is the product of the travel time t_e and the (per-unit) delay price λ. Then for a given delay price λ, each edge selects the optimal travel time to minimize its *generalized* cost, including both fuel cost $c_e(t_e)$ and delay cost λt_e. Thus, $t_e^*(\lambda)$ is the *best response* of edge e for a given delay price λ.

In equality (E_4), $w_e(\lambda)$ is defined as

$$w_e(\lambda) \triangleq c_e(t_e^*(\lambda)) + \lambda t_e^*(\lambda), \tag{15}$$

which can be interpreted as the minimal generalized cost (including both fuel cost and delay cost) of edge e for a given delay price λ. Obviously, $w_e(\lambda)$ is the generalized cost under the best response $t_e^*(\lambda)$.

In equality (E_5), since $\mathcal{X}$ restricts that an $s-d$ path is selected, $\min_{\boldsymbol{x}\in\mathcal{X}} \sum_{e\in\mathcal{E}} x_e \cdot w_e(\lambda)$ is exactly a shortest path problem where each edge e has a generalized cost $w_e(\lambda)$. We define $p^*(\lambda)$ as the resulting shortest-generalized-cost path.

In summary, (12) shows that for any dual variable λ, we only need to solve a shortest path problem to obtain the dual function value $D(\lambda)$, which is much easier than PASO.

4.3 The Heuristic Algorithm

Our heuristic scheme relies on one key observation. Define

$$\delta(\lambda) \triangleq \sum_{e\in p^*(\lambda)} t_e^*(\lambda), \tag{16}$$

which is the total travel time of the resulting shortest-generalized-cost path $p^*(\lambda)$ for a given λ. Our key observation is the following theorem (see an example in Fig. 6).

Theorem 3. *$\delta(\lambda)$ is non-increasing over $\lambda \in [0, +\infty)$.*

Theorem 3 shows that increasing λ will decrease the total travel time of the selected path based on the best responses of all edges. Intuitively, since λ can be interpreted as a delay price, increasing λ will force all edges to select a shorter travel time and further force the resulting shortest-generalized-cost path to have a shorter travel time.

Based on Theorem 3, we can use a simple dual variable λ to *coordinate* the total travel time. For example, when $\delta(\lambda) > T$, we can increase λ such that $\delta(\lambda)$ can be decreased to finally satisfy the hard delay requirement. On the other hand, when $\delta(\lambda) < T$, it means that the truck travels very fast and there still exists some room to increase the travel time and thus decrease the fuel consumption. Then we decrease λ such that $\delta(\lambda)$ can be increased to reach T. This is called a *coordination mechanism* [20, Ch. 5.1.6]. Therefore, we aim to find a λ_0 such that $\delta(\lambda_0) = T$. However, our problem PASO is not convex but has a combinatorial difficulty. Thus it is not guaranteed to find such a λ_0. We thus call our binary search for λ_0 (Algorithm 4) as a heuristic scheme.

Algorithm 4 A Heuristic Scheme

```
 1: Set $\lambda_L = 0$
 2: Set $\lambda_U = \lambda_{\max}$
 3: while $\lambda_U - \lambda_L > \mathsf{tol}$ do
 4:     Set $\lambda_0 = \frac{\lambda_L + \lambda_U}{2}$
 5:     Get $t_e^*(\lambda_0)$ from Lemma 6 for all $e \in \mathcal{E}$
 6:     Get $w_e(\lambda_0) = c_e(t_e^*(\lambda_0)) + \lambda_0 t_e^*(\lambda_0)$ for all $e \in \mathcal{E}$
 7:     Get the shortest path $p^*(\lambda_0)$ in terms of $w_e(\lambda_0)$
 8:     if $\delta(p^*(\lambda_0)) = T$ then
 9:         return $(p^*(\lambda_0), \{t_e^*(\lambda_0)\})$
10:     else if $\delta(p^*(\lambda_0)) > T$ then
11:         Set $\lambda_L = \lambda_0$
12:         Set $p^*(\lambda_L) = p^*(\lambda_0)$
13:         Set $t_e^*(\lambda_L) = t_e^*(\lambda_0), \forall e \in \mathcal{E}$
14:     else
15:         Set $\lambda_U = \lambda_0$
16:         Set $p^*(\lambda_U) = p^*(\lambda_0)$
17:         Set $t_e^*(\lambda_U) = t_e^*(\lambda_0), \forall e \in \mathcal{E}$
18:     end if
19: end while
20: return $(p^*(\lambda_L), \{t_e^*(\lambda_L)\})$ and $(p^*(\lambda_U), \{t_e^*(\lambda_U)\})$
```

In Algorithm 4, we first set an initial lower bound $\lambda_L = 0$ and an initial upper bound $\lambda_U = \lambda_{\max}$ for the targeted λ_0. In practice, since we are considering the fuel consumption and λ can be interpreted as a delay price, $\lambda_{\max}$ can be reasonably set to be an upper bound of the fuel consumption per hour. In our simulation in Sec. 5, we set $\lambda_{\max} = 100$, which works for all settings. Then we do binary search in lines 3-19, where tol in line 3 is the tolerance level for termination which is close to zero. During the binary search, based on the non-increasing property of $\delta(\lambda)$ (Theorem 3), we keep updating the lower bound λ_L and its corresponding solution $(p^*(\lambda_L), \{t_e^*(\lambda_L) : e \in p^*(\lambda_L)\})$, as well as the upper bound λ_U and its corresponding solution $(p^*(\lambda_U), \{t_e^*(\lambda_U) : e \in p^*(\lambda_U)\})$.

This algorithm has two possible results:

▷ **Case 1:** If it returns in line 9, then we have found a λ_0 such that $\delta(\lambda_0) = T$. We prove that the returned solution is optimal for PASO in Theorem 4.

▷ **Case 2:** If it returns in line 20, then we have found a λ_0 such that $\delta(\lambda_L) > T$ and $\delta(\lambda_U) < T$. With a small enough tolerance level tol, $\lambda_L = \lambda_0 - \mathsf{tol}/2 \to \lambda_0^-$. Likewise, $\lambda_U = \lambda_0 + \mathsf{tol}/2 \to \lambda_0^+$. Roughly speaking, this means that $\delta(\lambda)$ is not continuous at $\lambda = \lambda_0$. Although this return does not guarantee optimality, we prove in Theorem 5 that the returned solutions $(p^*(\lambda_L), \{t_e^*(\lambda_L) : e \in p^*(\lambda_L)\})$ and $(p^*(\lambda_U), \{t_e^*(\lambda_U) : e \in p^*(\lambda_U)\})$ give a lower bound LB and an upper bound UB for OPT, respectively.

Theorem 4. *If Algorithm 4 returns in line 9, then the returned solution $(p^*(\lambda_0), \{t_e^*(\lambda_0) : e \in p^*(\lambda_0)\})$ is an optimal solution of PASO.*

As a by-product, Theorem 4 also shows that the strong duality for the combinatorial problem PASO holds in this case, and λ_0 is the optimal dual solution to PASO-Dual.

Theorem 5. *If Algorithm 4 returns in line 20, and define* $\mathsf{LB} \triangleq \sum_{e\in p^*(\lambda_L)} c_e(t_e^*(\lambda_L))$ *and* $\mathsf{UB} \triangleq \sum_{e\in p^*(\lambda_U)} c_e(t_e^*(\lambda_U))$*, then we have* $\mathsf{LB} \leq \mathsf{OPT} \leq \mathsf{UB}$.

The LB and UB returned by Algorithm 4 in line 20 can be used for *Step 1* of Algorithm 3. For the case that Algorithm 4 returns in line 9, we use the returned optimal solution as both a lower bound and an upper bound with LB = UB = OPT. After such unification, Algorithm 4 always outputs a LB and UB for the optimal solution OPT.

Time Complexity: If we use Dijkstra's shortest-path algorithm with a min-priority queue in line 7 in Algorithm 4, Algorithm 4 has complexity $O((m+n\log n)\log\lambda_{\max})$, much faster than the FPTAS (Algorithm 3).

Remark: A similar dual-based heuristical approach for RSP is proposed in [31]. However, as mentioned in Sec. 3, different from RSP, our problem PASO has an extra design space of speed optimization. Therefore, theoretically our contribution in this section is to generalize the dual-based heuristical design from RSP [31] to PASO.

5. PERFORMANCE EVALUATION

In this section, we use real-world data to evaluate the performance of our algorithms. Our objectives are three-fold: (i) collect realistic dataset and model the fuel-rate-speed function, (ii) evaluate and compare the performance of our FPTAS and heuristic, and (iii) compare our algorithms with baseline algorithms, including both shortest path algorithm and fastest path algorithm adapted from common practice.

5.1 Dataset

Transportation Network: We construct the U.S. National Highway Systems (NHS) from the dataset of Clinched Highway Mapping (CHM) Project [42]. The whole highway network graph file is specified in [1], which consists of 84504 nodes (waypoints) and 89119 (one-direction) edges.

Elevation: In this paper, we consider the grade/slope effect when modeling the road-dependent fuel-rate-speed function. To obtain the road grades, we use the Elevation Point Query Service [9] provided by the U.S. Geological Survey (USGS) to query elevations of all nodes in the NHS graph.

Speed Limits: We use the historical average speed as the maximal speed R_e^{ub} for each road e. HERE map [7] has put speed detectors over many countries including U.S., and it provides APIs to query location-based real-time speed information. We collect the real-time speed information from HERE map [7] for two weeks and use the average as R_e^{ub} for each road e in the NHS graph. For the minimal speed limit R_e^{lb}, we manually set it to be $R_e^{\mathsf{lb}} = \min\{30, R_e^{\mathsf{ub}}\}$.

Fuel Consumption Data: It is hard for us to get suitable real-world fuel consumption data. In this paper, we instead leverage the widely-used ADVISOR simulator [37] to collect fuel consumption data (see Sec. 5.2).

Heavy-Duty Truck: Fuel consumption highly depends on the truck type. Another benefit of using ADVISOR is that it also provides some heavy-duty truck configurations. In this simulation, we use the Kenworth T800 Vehicle [5], a Class 8 heavy-duty truck, with 36-ton full load. It is specified in files `VEH_KENT800Trailer.m` and `HeavyTruck_in.m` in ADVISOR with the following parameters in Tab. 1.

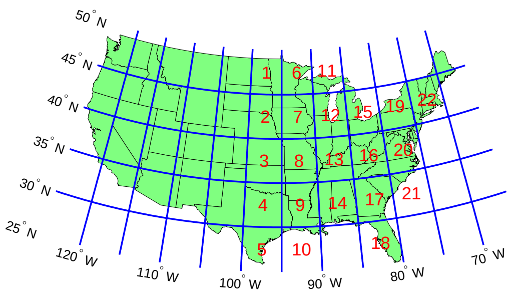


Figure 4: U.S. map and 22 regions.

Table 1: Truck Parameters (Kenworth T800).

Drag Coefficient c_d	Frontal area A_f	Glider Mass	Cargo Mass
0.7	8.5502 m^2	2,552kg	33,234kg

Preprocessing Highway Network: In the original NHS graph from CHM [1], we observe that: (i) most roads are in the "eastern" U.S., and (ii) many roads are very short with degree-1 endpoints (non-intersection roads). To create a network with more diverse paths, we first cut the whole NHS graph to the "eastern" part with longitude to the east of $100°W$ (see Fig. 4). We further merge the non-intersection roads with the same level of grades into a single road. Some network statistics after these two kinds of preprocessing are shown in Tab. 2. Note that since the average distance for each edge is 3.26 miles after preprocessing, it is reasonable to ignore the speed transition over two adjacent edges, which justifies the assumption in our fuel consumption model.

Moreover, to better visualize our results, we divide the major "eastern" U.S. into 22 regions (see Fig. 4). In each region $i \in [1, 22]$, we find the node in the graph which is nearest to the region's center. We also call it node i. Next we will use such 22 nodes as the source and destination nodes.

Table 2: Network Statistics. "O" is the original NHS graph, "E" is the "eastern" graph (to the east of $100°W$), and "M" is the merged one. θ is the grade.

$\mathcal{G}$	n	m	avg D_e (mile)	avg R_e^{lb} (mph)	avg R_e^{ub} (mph)	avg $\lvert\theta\rvert$ (%)
O	84504	178238	2.08	37.4	55.97	0.64
E	65520	137521	1.97	37.3	55.55	0.58
M	38213	82781	3.26	36.43	54.19	0.82

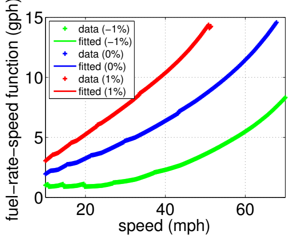
5.2 Model Fuel-Rate-Speed Function

We model the fuel-rate-speed function as

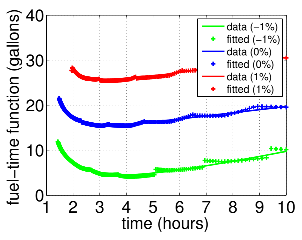
$$f_e(x) = a_e x^3 + b_e x^2 + c_e x + d_e, \forall e \in \mathcal{E} \tag{17}$$

Here x is the speed (unit: mph) and $f_e(x)$ is the fuel rate consumption (unit: gph (gallons per hour)). Although our model (17) can capture any road-dependent features/factors, e.g., grade, rolling resistance, and air density, etc., we only consider the road grade θ in this simulation, which is the major factor for truck fuel consumption [14]. We collect fuel consumption data from ADVISOR and fit fuel-rate-speed functions $f_e(x)$ by MATLAB's `fit` tool. Due to the space limitation, the details are shown in our technical report [24].

Our results show that the fuel-rate-speed function $f_e(x)$ is strictly convex in reasonable speed limit regions. More concretely, we visualize the fuel-rate-speed function $f_e(x)$ and fuel-time function $c_e(t_e)$ for three sampled grades, -1.0%, 0.0%, and 1.0%, as shown in Fig. 5. We can see that both of them are strictly convex in reasonable regions. We also verify that $c_e(t_e)$ will first strictly decrease and then strictly increasing and thus we only need to focus on the decreasing interval without loss of optimality, as discussed in Sec. 2.2.



(a) Fuel-rate-speed function $f_e(x)$.



(b) Fuel-time function $c_e(t_e)$ over a 100-mile road.

Figure 5: Fit curve v.s. data for grades 0%, ±1%.

5.3 Evaluate/Compare FPTAS and Heuristic

We implement our algorithms with C++ where we use the SNAP graph structure [33]. We evaluate on a server with an 8-core Intel Core-i7 3770 3.4 Ghz CPU and 16 GB memory, running CentOS 6.4. To evaluate and compare our FPTAS (Algorithm 3) and heuristic scheme (Algorithm 4), we consider 4 different settings, S1, S2, S3, and S4, as shown in Tab. 3. Note that since we aim to compare them, we use LB = 1 and UB = 1000 in *Step 1* of Algorithm 3.

In terms of the minimized fuel cost of the algorithms, Tab. 3 shows that the heuristic scheme always outputs the optimal solution (LB = UB, hence LB = UB = OPT), and the FPTAS also outputs a near-optimal solution (e.g., in S1, 74.812 is only a little bit larger than OPT = 74.811). This demonstrates that both FPTAS and the heuristic scheme have good performance. However, in terms of time/space complexity, the heuristic scheme is much better than FPTAS. As we can see, the FPTAS only works fine for the small-scale settings (S1 and S4), where the transportation network in regions 1 and 2 in Fig. 4 is considered, with only 1185 nodes and 2568 edges. When we use a little bit larger scale setting S2, it runs for nearly 1 hour and consumes 14.76 GB memory (out of 16 GB in total). Our server cannot run any other setting whose scale is larger than S2. We also note that the complexity of the FPTAS increases significantly as we decrease ϵ from 0.1 to 0.05, as shown in settings S1&S4. Contrarily, our heuristic scheme can handle all 22 regions (setting S3) with 38213 nodes and 82781 edges easily with low time/space complexity.

Tab. 3 verifies that the FPTAS is not necessarily scalable to practical large-scale highway networks, but our heuristic scheme works very well in terms of both performance and complexity. To see why the heuristic scheme performs well, we examine an example source-destination pair in the setting S3, $(s, d) = (4, 22)$, and plot its $\delta(\lambda)$ function (the total travel time of the shortest-generalized-cost path, see (16)) in Fig. 6. We observe that function $\delta(\lambda)$ is non-increasing, which verifies Theorem 3. Moreover, $\delta(\lambda)$ has only a few small non-continuous jumps (e.g., a jump at point $\lambda = 11.47$ from 37.83 to 37.62). Whenever a (feasible) delay is not within such jump regions, we can always find a λ_0 such that $\delta(\lambda_0) = T$. According to Theorem 4, the output solution must be optimal. For example, when $T = 40$, we can find $\lambda_0 = 4.48$ such that $\delta(\lambda_0) = 40$, as shown in Fig. 6. The optimal solution can be derived as $(p^*(\lambda_0), \{t_e^*(\lambda_0)\})$. Even when T is within one of such jump regions (e.g., $T \in (37.62, 37.83)$), since the length of the delay region (e.g., $(37.62, 37.83)$ has a length of 0.21 hours) is often negligible as compared to a nearly 40-hour travel, the output LB and UB would be very close. Hence, our heuristic scheme outputs an optimal (at least near-optimal) solution for any input T. We will further justify this observation with more instances in Sec. 5.4.

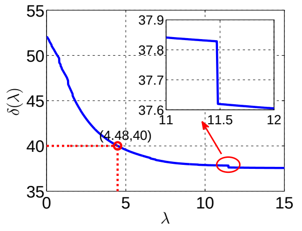


Figure 6: An example for $\delta(\lambda)$ when $(s, d) = (4, 22)$.

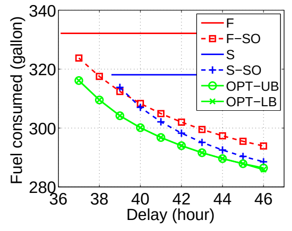


Figure 7: The delay effect when $(s, d) = (9, 22)$.

Table 4: Description of 6 solutions.

Solution	Description	Benchmark
F	Sol. of fastest path with maximal speed	Time
F-SO	Sol. of fastest path with optimal speed	-
S	Sol. of shortest path with maximal speed	Distance
S-SO	Sol. of shortest path with optimal speed	Distance
OPT-LB	Sol. of LB of our heuristic scheme	Fuel
OPT-UB	Sol. of UB of our heuristic scheme	-

5.4 Compare Performance with Baselines

In this section, we compare the performance of our heuristic scheme with the following 4 baseline algorithms: (i) fastest (time) path algorithm with maximal speed, (ii) fastest path algorithm with optimal speed, (iii) shortest (distance) path algorithm with maximal speed, and (iv) shortest path algorithm with optimal speed. Each of them outputs one solution for PASO. Since our heuristic scheme outputs two solutions respectively corresponding to the LB and UB, we have 6 solutions in total, as summarized in Tab. 4.

In later comparison, since the travel time of F is the minimal time for any feasible solution of PASO, we will use it as the *time benchmark*. For example, a solution SOL (e.g., SOL could be OPT-UB) with time increment 10% means that $\frac{\text{Travel time of SOL} - \text{Travel time of F}}{\text{Travel time of F}} = 10\%$. Similarly, we use the travel distance of S/S-SO as the *distance benchmark*, and use the fuel consumption of OPT-LB as the *fuel benchmark*.

Now we input all 22 regions in Fig. 4 as the underlaying highway network and use all permutations of the 22 nodes (the nearest points to each individual region) as (s, d) pairs. For each (s, d) pair, we use ten different delays, from $\lceil T^f \rceil$ to $\lceil T^f \rceil + 9$ where T^f is the fastest travel time from s to d.

A Single Instance: We first consider one instance $(s, d, T) = (9, 22, 40)$. Tab. 5 compares the 6 solutions As we can see, our heuristic scheme again outputs the optimal solution. It consumes 300.1 gallons of fuel, runs 10.76% slower than

Table 3: Comparisons of FPTAS and Heuristic. Here an instance is the tuple (source, destination, delay), i.e., (s, d, T). For example, in S1, (1,2,8) means that the source (resp. destination) node is 1 (resp. 2), which is the nearest node to the center of region 1 (resp. region 2) in Fig. 4, and the total delay is 8 hours.

No.	Network			Input		Performance (gallon)		Time (second)		Memory (GB)	
	Reg.	n	m	Instance	ϵ	Heuri. LB/UB	FPTAS	Heuri.	FPTAS	Heuri.	FPTAS
S1	1&2	1185	2568	(1,2,8)	0.1	74.811/74.811	74.812	1	50	0.29	2.73
S2	17&18	3274	7465	(18,17,10)	0.1	60.2795/60.2795	60.2798	2	3511	0.29	14.76
S3	1-22	38213	82781	(4,22,40)	0.1	290.744/290.744	-	365	-	0.29	-
S4	1&2	1185	2568	(1,2,8)	0.05	74.811/74.811	74.812	1	126	0.29	6.84

Table 5: Performance of instance $(s, d, T) = (9, 22, 40)$.

Sol.	Time (hour)	Incre. (%)	Dist. (mile)	Incre. (%)	Fuel (gal.)	Incre. (%)
F	36.11	-	1821	2.71	332.1	10.67
F-SO	40	10.76	1821	2.71	308.3	2.73
S	38.58	6.85	1773	-	318.0	5.99
S-SO	40	10.76	1773	-	307.0	2.30
OPT-LB	40	10.76	1778	0.30	300.1	-
OPT-UB	40	10.76	1778	0.30	300.1	0

Table 6: Average performance of all instances.

Sol.	Avg Time Incre.(%)	Avg Dist. Incre.(%)	Avg Fuel Incre.(%)	Avg Fuel Econ.(mpg)
F	-	1.71	20.14	5.05
F-SO	32.80	1.71	2.00	5.94
S	2.82	-	16.40	5.13
S-SO	32.80	-	0.31	5.94
OPT-LB	32.95	0.17	-	5.96
OPT-UB	32.89	0.18	0.02	5.96

the time benchmark (F), and 0.3% longer than the distance benchmark (S/S-SO). Also, without speed optimization, the fastest path (F) consumes 32 more gallons (10.67%) and the shortest path (S) consumers 18 more gallons (5.99%). But with speed optimization, both fastest path and shortest path have near-optimal performance.

For $(s, d) = (9, 22)$, we also evaluate the effect of input delay T as shown in Fig. 7. Considering speed optimization, when the input delay $T \in [36.11, 38.58)$, the shortest path is infeasible, which shows that fastest path outperforms shortest path. The shortest path becomes feasible when $T \geq 38.58$, and it outperforms the fastest path when $T > 39$. This figure thus shows that the shortest path becomes better and better as the delay constraint increases. Intuitively, when the hard delay constraint can be satisfied, the travel distance would be critical for the total fuel consumption.

The OPT-UB curve in Fig. 7 is the *energy-delay tradeoff* of $(s, d) = (9, 22)$. We see that increasing delay can save fuel consumption, and the saving has a "diminishing" property. For example, the truck can save 6.6 gallons of fuel if it increases its delay from 37 to 38 hours, but the saving reduces to 1.46 gallons if its delay is relaxed from 45 to 46 hours.

All Instances: Similar to Tab. 5, we can get the time, distance, and fuel of the 6 solutions for all source-sink pairs. We evaluate the average performance of all running instances in terms of time/distance/fuel increments compared to the benchmark numbers, as summarized in Tab. 6. Note that in 4.84% of instances, shortest path is infeasible. Tab. 6 only has the average performance over the instances where the shortest path is feasible.

Tab. 6 shows that on average OPT-UB only consumes 0.02% of more fuels than the fuel benchmark (OPT-LB). This again shows that our heuristic scheme outputs a near-optimal solution in all instances.

For the baseline algorithms, Tab. 6 shows that the fastest path (resp. shortest path) algorithm without speed optimization consumes 20.14% (resp. 16.40%) of more fuels than our solution. Our heuristic solution also improves the *36-ton*-truck's fuel economy from 5.05 for the fastest path and 5.13 for the shortest path to 5.96. Considering its significant portion of energy consumption, our solution can indeed save much fuel cost for the long-haul heavy-duty trucks.

When we allow speed optimization for the fastest path and the shortest path, we find that on average both of them are close to the optimal solution. More specifically, F-SO consumes 2.00% of more fuels and S-SO only consumes 0.31% of more fuels than OPT-LB. This apparently suggests that in the U.S., it is good enough to first choose the shortest or fastest path and then do speed optimization. However, in our simulation, the shortest path is infeasible among 4.84% of all instances, and the fastest path with speed optimization can consume 21.32% of more fuels in the worst instance. As opposed to them, our PASO solution is robust in the sense that it always output a solution that is both feasible and near-optimal. We also leave it as a future work to understand under which conditions the fastest/shortest path with speed optimization is close to the optimal solution.

6. CONCLUSION AND FUTURE WORK

Provisioning both energy-efficient and timely delivery is of great importance for logistic operators. This paper presents a first step to study the energy-efficient timely transportation problem with an emphasis for long-haul heavy-duty trucks. We propose two algorithms: the first one is an FPTAS and the second one is a heuristic with lower complexity and near-optimal empirical performance. Our real-world data-driven simulations show that our solution guarantees timely delivery and can save up to 17% of fuel consumption as compared to a fastest/shortest path algorithm adapted from common practice. An interesting and important future direction is to generalize our results beyond the highway setting to cover more sophisticated local driving scenarios.

Acknowledgment

The work presented in this paper was supported in part by National Basic Research Program of China (Project No. 2013CB336700) and the University Grants Committee of the Hong Kong Special Administrative Region, China (Theme-based Research Scheme Project No. T23-407/13-N and Collaborative Research Fund No. C7036-15G).

7. REFERENCES

[1] CHM U.S. national highway systems. http://courses.teresco.org/chm/graphs/usa-national.gra.

[2] Energy consumption estimates by end-use sector, ranked by state, 2013. http://www.eia.gov/state/seds/data.cfm?incfile=/state/seds/sep_sum/html/rank_use.html&sid=US.

[3] Fuel economy at various driving speeds. http://www.afdc.energy.gov/data/10312.

[4] Keeping your vehicle in shape. https://www.fueleconomy.gov/feg/maintain.jsp.

[5] Kenworth T800 vehicle. http://www.kenworth.com/trucks/t800.

[6] Place an order with guaranteed delivery. https://www.amazon.com/gp/help/customer/display.html/ref=hp_left_v4_sib?ie=UTF8&nodeId=201117390.

[7] Traffic flow using corridor in HERE maps. https://developer.here.com/api-explorer/rest/traffic/flow-using-corridor.

[8] Transportation overview. http://www.c2es.org/energy/use/transportation.

[9] USGS elevation point query service. http://nationalmap.gov/epqs/.

[10] Smarter trucking saves fuel over the long haul. http://news.nationalgeographic.com/news/energy/2011/09/110923-fuel-economy-for-trucks/, 2011.

[11] Improving the fuel efficiency of American trucks. https://www.whitehouse.gov/the-press-office/2014/02/18/fact-sheet-opportunity-all-improving-fuel-efficiency-american-trucks-bol, 2014.

[12] Transportation logistics enhances your business efficiency. http://www.readytrucking.com/transportation-logistics-business-efficiency/, 2014.

[13] Supertruck team achieves 115% freight efficiency improvement in Class 8 long-haul truck. http://energy.gov/eere/vehicles/articles/supertruck-team-achieves-115-freight-efficiency-improvement-class-8-long-haul, 2015.

[14] K. Ahn. Microscopic fuel consumption and emission modeling. Master's thesis, Virginia Polytechnic Institute and State University, 1998.

[15] A. A. Alam, A. Gattami, and K. H. Johansson. An experimental study on the fuel reduction potential of heavy duty vehicle platooning. In *Prof. IEEE ITSC*, 2010.

[16] F. An and M. Ross. Model of fuel economy with applications to driving cycles and traffic management. *Transportation Research Record*, 1993.

[17] S. Ardekani, E. Hauer, and B. Jamei. Traffic impact models. *Chapter 7 in Traffic Flow Theory, Oak Bridge National Laboratory Report*, 1992.

[18] B. H. Ashby. *Protecting Perishable Foods during Transport by Truck*. U.S. Department of Agriculture, 2006.

[19] I. M. Berry. The effects of driving style and vehicle performance on the real-world fuel consumption of U.S. light-duty vehicles. Master's thesis, Massachusetts Institute of Technology, 2010.

[20] D. P. Bertsekas. *Nonlinear Programming*. Athena scientific, 1999.

[21] S. C. Davis, S. W. Diegel, and R. G. Boundy. *Transportation Energy Data Book: Edition 34*. U.S. Department of Energy, 2015.

[22] E. Demir, T. Bektaş, and G. Laporte. A comparative analysis of several vehicle emission models for road freight transportation. *Transportation Research Part D: Transport and Environment*, 2011.

[23] E. Demir, T. Bektaş, and G. Laporte. A review of recent research on green road freight transportation. *European Journal of Operational Research*, 2014.

[24] L. Deng, M. H. Hajiesmaili, M. Chen, and H. Zeng. Energy-efficient timely transportation of long-haul heavy-duty trucks. Technical Report, http://www.ie.cuhk.edu.hk/%7Emhchen/papers/EETT.eEnergy.16.TR.pdf.

[25] W. Ford Torrey and D. Murray. An analysis of the operational costs of trucking: A 2015 update. 2015.

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

[27] W. Harrington and A. Krupnick. Improving fuel economy in heavy-duty vehicles. *Resources for the Future DP*, 2012.

[28] R. Hassin. Approximation schemes for the restricted shortest path problem. *Mathematics of Operations Research*, 1992.

[29] E. Hellström, J. Åslund, and L. Nielsen. Design of an efficient algorithm for fuel-optimal look-ahead control. *Control Engineering Practice*, 2010.

[30] E. Hellström, M. Ivarsson, J. Åslund, and L. Nielsen. Look-ahead control for heavy trucks to minimize trip time and fuel consumption. *Control Engineering Practice*, 2009.

[31] A. Jüttner, B. Szviatovski, I. Mécs, and Z. Rajkó. Lagrange relaxation based method for the QoS routing problem. In *Proc. IEEE INFOCOM*, 2001.

[32] J. Larson, K.-Y. Liang, and K. H. Johansson. A distributed framework for coordinated heavy-duty vehicle platooning. *IEEE Transactions on Intelligent Transportation Systems*, 2015.

[33] J. Leskovec and R. Sosič. SNAP: A general purpose network analysis and graph mining library in C++. http://snap.stanford.edu/snap, 2014.

[34] D. H. Lorenz and D. Raz. A simple efficient approximation scheme for the restricted shortest path problem. *Operations Research Letters*, 2001.

[35] W. Mallett. *Freight Performance Measurement: Travel Time in Freight-Significant Corridors*. U.S. Federal Highway Administration, 2006.

[36] F. Mannering, W. Kilareski, and S. Washburn. *Principles of Highway Engineering and Traffic Analysis*. John Wiley & Sons, 2007.

[37] T. Markel, A. Brooker, T. Hendricks, V. Johnson, K. Kelly, B. Kramer, M. O'Keefe, S. Sprik, and K. Wipke. ADVISOR: a systems analysis tool for advanced vehicle modeling. *Journal of Power Sources*, 2002.

[38] Z. Mohamed-Kassim and A. Filippone. Fuel savings on a heavy vehicle via aerodynamic drag reduction. *Transportation Research Part D: Transport and Environment*, 2010.

[39] E. K. Nam and R. Giannelli. Fuel consumption

modeling of conventional and advanced technology vehicles in the physical emission rate estimator (PERE). *U.S. Environmental Protection Agency*, 2005.

[40] F. Stodolsky, L. Gaines, and A. Vyas. Analysis of technology options to reduce the fuel consumption of idling trucks. Technical report, Argonne National Lab, 2000.

[41] Y. Suzuki. A new truck-routing approach for reducing fuel consumption and pollutants emission. *Transportation Research Part D: Transport and Environment*, 2011.

[42] J. D. Teresco. The Clinched Highway Mapping (CHM) project. http://cmap.m-plex.com/.

[43] M. Tunnell. Estimating truck-related fuel consumption and emissions in maine: A comparative analysis for six-axle, 100,000 pound vehicle configuration. In *Proc. TRB Annual Meeting*, 2011.