

Article

The Shortest Path Problems in Battery-Electric Vehicle Dispatching with Battery Renewal

Minfang Huang ^{1,*} and Jing-Quan Li ²¹ School of Economics and Management, North China Electric Power University, Beijing 102206, China² California PATH, University of California, Berkeley, Richmond, CA 94804, USA; jingquan.li@gmail.com

* Correspondence: huangmf@ncepu.edu.cn; Tel.: +86-10-6177-3126

Academic Editor: Andrew Kusiak

Received: 26 April 2016; Accepted: 22 June 2016; Published: 28 June 2016

Abstract: Electric vehicles play a key role for developing an eco-sustainable transport system. One critical component of an electric vehicle is its battery, which can be quickly charged or exchanged before it runs out. The problem of electric vehicle dispatching falls into the category of the shortest path problem with resource renewal. In this paper, we study the shortest path problems in (1) electric transit bus scheduling and (2) electric truck routing with time windows. In these applications, a fully-charged battery allows running a limited operational distance, and the battery before depletion needs to be quickly charged or exchanged with a fully-charged one at a battery management facility. The limited distance and battery renewal result in a shortest path problem with resource renewal. We develop a label-correcting algorithm with state space relaxation to find optimal solutions. In the computational experiments, real-world road geometry data are used to generate realistic travel distances, and other types of data are obtained from the real world or randomly generated. The computational results show that the label-correcting algorithm performs very well.

Keywords: electric vehicles; battery swapping or charging; resource-constrained shortest path; label-correcting algorithm

1. Introduction

Recently, aimed at reducing vehicle emissions and greenhouse gases, more and more electric transit buses and delivery trucks have been used in personal travel, public transit, home deliveries from grocery stores, postal deliveries and courier services. For example, electric motorcycles or scooters have been promoted through incentive programs in Taiwan. UPS began using electric vehicles in their fleet as early as 1933, and now almost 2000 electric delivery trucks have been used [1]. More than 80 electric transit buses were used in the World Expo 2010 Shanghai [2]. As a benchmark city of electric vehicle demonstration application in China, Beijing released the Beijing electric vehicle promotion application action plan (2014–2017) in June, 2014 [3]. Although electric vehicles produce zero emissions, a fully-charged battery only allows running a limited operational distance. The limited short range may not be a problem for commute drivers in an urban area. It will have great impacts on the applications of vehicle scheduling and routing. In order to make electric vehicles service for a whole day, the battery near depletion can be either quickly charged or exchanged with a fully-charged one at a battery service station. Therefore, the problem of electric vehicle dispatching with battery renewal falls into the category of the resource-constrained shortest path problem.

The resource-constrained shortest path problem consists of determining the lowest-cost path from an origin node to a destination node on a network, such that resources consumed along the path are no more than specified values. Some typical examples of resources include maximum daily working hours, vehicle capacity, fuel tank size and maximum traveled distances. The resource-constrained shortest path problem has significant applicants in transportation, such as in vehicle scheduling with

time windows [4], vehicle routing with time windows [5], transit crew scheduling [6] and airline crew pairing [7]. In these applications, the resource-constrained shortest path problem often appears as a sub-problem in the context of column generation. One critical component of an electric vehicle is its battery. Unlike the resources previously mentioned, a battery can be quickly charged or exchanged with a fully-charged one before it runs out. Such an aspect results in the shortest path problem with resource renewal, which will be investigated in this paper.

The resource-constrained shortest path problem may be classified into two types. In the first type, either the problem is based on an acyclic network, or although the underlying network is cyclic, there are no negative cost cycles (e.g., all arcs have positive costs). Solution approaches include branch-and-bound [8], label setting [9], label setting with preprocessing [10], K-shortest path enumeration [11,12] and Lagrangian relaxation with robust shortest path enumeration [13]. In the second type, there are negative cost cycles in the underlying network. If no cycles are allowed in the shortest path, such a problem is called the elementary shortest path problem with resource constraints. Solution approaches include branch-and-bound [8], label correcting [14], bi-directional label correcting [15] and label correcting with state space relaxation [16]. A relaxed version allows the existence of cycles in the shortest path. Solution methods are mainly based on dynamic programming with the strategies of two-cycle elimination [5,17], k-cycle elimination [18] and a combined (k, 2)-loop elimination [19]. Both types of resource-constrained shortest path problems are NP-hard [20]. The literature about the electric vehicle routing problem is focused on refueling station location [21,22], battery swapping or charging replenishment [23,24] and the electric traveling salesman problem [25] or the electric vehicle routing problem [26–28].

In this paper, we consider the shortest path problems in two types of electric vehicles with battery renewal: electric buses and electric trucks. The problem on electric buses is defined as follows: given an origin and a destination depot, a set of scheduled trips with starting and ending times, the travel distance between two locations, the maximal range of the electric bus before battery renewal and the cost between two nodes, find the shortest path from the origin depot to the destination depot. The problem on electric trucks is similar, but with additional constraints: the service has flexible time windows and demand, and the truck has limited capacity.

The contributions of this paper are as follows. First, we model the shortest path problem in two important applications: (1) electric transit bus scheduling and (2) electric truck routing with time windows. We propose integer programming-based formulations. Second, we develop a dynamic programming-based algorithm to find optimal solutions for these emerging shortest path problems. Third, we conduct the extensive computational experiments in the context of real-world settings: the travel distance is obtained from real-world road geometry data; the battery service time and operational distance are obtained from real-world experiences; and the unit electricity cost is estimated based on real-world testing reports. We also compare the computational time with CPLEX.

The rest of this paper is organized as follows. Section 2 models the shortest path problems with battery renewal. Section 3 presents the label-correcting algorithm. Numerical studies are presented in Section 4. Section 5 concludes the paper and describes future research directions.

2. The Shortest Path Problems in Electric Vehicle Dispatching

In this section, we first model and formulate the shortest path problem in electric transit bus scheduling with battery renewal, where the underlying network is acyclic. We then model and formulate the shortest path problem in electric truck routing with battery renewal and time windows, where the underlying network may contain negative cost cycles.

2.1. The Shortest Path Problem in Electric Transit Bus Scheduling

Heavy-duty diesel trucks and buses generate more than 45% of the nitrogen oxide and 75% of the particulate matter [29]. In contrast, electric transit buses have a unique advantage: zero emissions. The use of electric buses in transit service may date back to the 1990s in Santa Barbara, California [30],

and Chattanooga, Tennessee [31], with the main purpose of reducing vehicle emissions in downtown areas. Electric buses were used at a large scale during the Beijing Summer Olympics [32] and World Expo 2010 Shanghai [2]. It was reported in [33] that nine electric buses have been used in Seoul. In Los Angeles, Foothill Transit has already used a number of electric buses in service with a future goal of a 100% clean-fuel fleet [34].

A major problem of electric buses is the restricted operational distance caused by the limited battery capacity. For example, a bus with 22 to 25 seats can run 60 to 90 km in Chattanooga before the battery runs out [31]. In the Beijing Olympics, the maximum operational distance is 130 km [32]. The electric buses servicing Seoul can run 80 km [33]. In the Foothill Transit of Los Angeles, a fully-charged battery can propel the bus for 3 h [34]. In [35], certain specifications of electric bus engines are listed. The use of an air conditioner has a great impact on the operational distance. The United Nations Environment Programme [2] reported that the maximum distance was 150 km with an air conditioner on and 250 km with an air conditioner off in the Shanghai World Expo. In order to overcome the short operational distance, a battery near depletion can be charged or exchanged with a fully-charged one at a battery service station. The fast charge is used in Seoul and Los Angeles; the charging time is from 20 to 30 min in Seoul [33] and about 10 min in Los Angeles [34]. Battery exchange has been employed in Tennessee [31], Beijing [30] and Shanghai [36]. The time of each battery exchange is 10 to 15 min in Tennessee [31].

Both battery charging and exchange are generally accomplished at a battery management station, which has limited service spaces. If a bus arrives at a battery station without any available space, the bus has to wait. We define a maximum waiting time at each battery station in order to avoid unnecessarily long waiting. The capacity of battery stations cannot be violated at any moment. We use a discretization approach to handle this requirement. For each trip, the earliest service start time at a battery station is the arrival time, and the latest service start time is the arrival time plus the maximum waiting time. We then sample the start time range and generate certain discretized nodes. The discretization is applied to all of the trips. The discretized nodes are called time-expanded station nodes, each of which corresponds to a battery service station at a specified time. Such a discretization technique has been widely used in the single-track train scheduling problem with limited spaces at a terminal (e.g., see [37]) and transit bus scheduling with the maximum route distance constraint for compressed natural gas, diesel or hybrid-diesel buses [38].

We now formally define the shortest path problem. The parameters that will be used in the paper are listed as follows.

- B the set of available battery stations;
- T_b the set of time-expanded station nodes associated with $b \in B$;
- $T = \bigcup_{b \in B} T_b$ the set of all time-expanded nodes;
- a_t service starting time;
- U_t the battery service (swapping or charging) time at node $t \in T$;
- $a_t + U_t$ service ending time;
- S the set of service trips;
- a_s starting time, $s \in S$;
- U_s the service duration, $s \in S$;
- o the origin depot;
- d the destination depot;
- D the maximum distance that a fully-charged battery allows;
- d_{ij} the travel distance from $i \in \{o \cup T\}$, or the ending point of trip $i \in S$ to $j \in \{d \cup T\}$, or the ending point of trip $j \in S$; the trip service distance is included if $j \in S$;
- t_{ij} the travel time from $i \in \{o \cup T\}$, or the ending point of trip $i \in S$ to $j \in \{d \cup T\}$, or the ending point of trip $j \in S$;
- A the set of arcs, including the following arcs: (1) arcs from o to $i \in S$; (2) arcs from $i \in S \cup T$ to d ; and (3) arcs from $i \in S \cup T$ to $j \in S \cup T$ if $a_i + U_i + t_{ij} \leq a_j$ and at least one of i and j belongs to S ;

$\bar{c}_{ji}$ the arc cost, often defined as the operational cost; if the shortest path appears as a sub-problem in column generation or Lagrangian relaxation, $\bar{c}_{ji}$ is defined as a function of the operational cost of arc ji and linear or Lagrangian dual variables; it is worthy to mention that the underlying network is acyclic because the starting times of all nodes in $S \cup T$ are fixed.

The decision variables are defined as follows.

x_{ji} a binary variable; $x_{ji} = 1$ if a vehicle is assigned to node i directly after node j ;

g_s a continuous variable that indicates the accumulative distance traveled to the ending point of trip $s \in S$ since the latest battery renewal.

The shortest path problem in electric transit bus scheduling with battery renewal is formulated in Table 1.

Table 1. The mathematical model of the shortest path problem in electric transit bus scheduling with battery renewal.

Objective Function and Restrictions	Variable Range	Equation No.
$\min \sum_{(j,i) \in A} \bar{c}_{ji} x_{ji}$		(a1)
st		
$\sum_{j:(i,j) \in A} x_{ij} = 1$	$\forall i = o$	(a2)
$\sum_{j:(j,i) \in A} x_{ji} = -1$	$\forall i = d$	(a3)
$\sum_{i:(j,i) \in A} x_{ji} - \sum_{i:(i,j) \in A} x_{ij} = 0$	$\forall j \in S \cup T$	(a4)
$g_i = \sum_{j:(j,i) \in A} (g_j + d_{ji}) x_{ji}$	$\forall i \in S$	(a5)
$g_i = 0$	$\forall i \in T \cup \{o\}$	(a6)
$(g_j + d_{jt}) x_{jt} \leq D$	$\forall t \in T \cup \{d\}, (j, t) \in A$	(a7)
$x_{ij} \in \{0, 1\}$	$(i, j) \in A$	

The objective is to minimize the total cost. Constraint (a2) ensures sending a unit flow; Constraint (a3) forces receiving a unit flow; and Constraint (a4) is flow conservation. Constraint (a5) determines the accumulative distance traveled since the latest battery renewal. Constraint (a6) sets the accumulative distance as zero at battery stations and the origin depot. Constraint (a7) forces that the maximum route distance restriction is satisfied.

If a sufficiently large penalty is imposed for each arc ji with $i \in T$, the problem becomes the resource-constrained shortest path problem, which is NP-hard [20]. Therefore, the shortest path problem in electric transit bus scheduling is NP-hard.

2.2. The Shortest Path Problem in Electric Truck Routing with Time Windows

The rapid development of online retailing has brought the huge demand of last-mile delivery of parcels. The application of electric trucks will offer great potential to reduce greenhouse gas emissions. FREVUE (Freight Electric Vehicles in Urban Europe) provides evidence on how innovative solutions using electric freight vehicles can help to achieve emission free city logistics. It reports that the electric vehicles should be used in a two-phase delivery network [25], which has been widely adopted by big online retailers, e.g., yhd.com and jd.com in China. In the network, conventional trucks first send parcels to delivery terminals, and later, the electric vehicles finish the last-mile deliveries to customers. A number of electric trucks have already been used in a major shipping company in order to reduce emissions and maintenance costs [1]. It was also reported in 2011 [39] that UPS agreed to purchase 100 electric vans for use in California. Major car manufacturers also produce electric shuttles now. For example, Ford Motor has proposed a type of electric van, named the Ford Transit Connect [40]. Electric delivery trucks also face the restriction of the maximum operational distance. For example, the maximum operational distance of Ford Transit Connect is 120 km [40], and the new trucks that UPS

agreed to buy can run about 140 km after a full battery charge [39]. It was reported that it can take six to eight hours for Ford Transit Connect to have a full charge [40].

In the current practice, electric trucks generally serve a relatively short and direct route [39] and do not have the problem of battery depletion. However, in order to use electric trucks at a large scale, the charging time has to be significantly reduced (e.g., using direct current charging) or a battery exchange station has to be built. Although we did not find reports on battery exchange for electric trucks, it can be expected that battery exchange facilities will be constructed in the future. Such construction may be benefited by financial incentives from governments and air-quality management districts. In fact, seven millions incentives were obtained when UPS agreed to buy electric vans [39]. We will assume that at least one battery station has been constructed for battery exchange or fast charging. Similarly, a battery service station can simultaneously serve a limited number of electric trucks.

We now define the shortest path problem formally.

S the set of the customers;

$[a_i, b_i]$ the time window of customer $i \in S$;

v_i demand of each node $i \in S$;

U_i service duration of each node $i \in S$;

D the maximum operational distance;

V the vehicle capacity;

d_{ij} the travel distance between each pair of locations;

t_{ij} the travel time between each pair of locations;

A the set of arcs, including the following arcs: (1) arcs from o to $i \in S$; (2) arcs from $i \in S \cup T$ to d ; and (3) arcs from $i \in S \cup T$ to $j \in S \cup T$ if $a_i + U_i + t_{ij} \leq b_j$ and at least one of i and j belongs to S ;

$\bar{c}_{ji}$ the arc cost, often defined as the distance cost and may include linear or Lagrangian dual variables if the shortest path appears as a sub-problem in column generation or Lagrangian relaxation;

B the set of available battery stations;

T_b the set of time-expanded station nodes associated with $b \in B$. This is determined in a similar way for electric bus scheduling: for each customer $i \in S$, we determine its earliest and latest service start time at a battery station. The earliest service start time equals a_i plus U_i plus the travel time to the battery station, while the latest service start time equals b_i plus U_i plus the travel time to the station plus the maximum waiting time. The discretization is then used to sample the start time range, and the time-expanded station nodes are generated. $T = \bigcup_{b \in B} T_b$ is then the set of all time-expanded nodes, each of which corresponds to a battery station with specific starting time $a_i = b_i$, $i \in T$.

Unlike the electric bus scheduling problem, the electric truck routing problem may be based on an underlying network with negative cost cycles if the width of time windows is large.

Decision variables are defined as follows.

s_i the service start time at node i ;

x_{ji}, g_s the same definition as in electric bus scheduling.

The shortest path problem in electric truck routing with the time windows, vehicle capacity and battery renewal is then formulated in Table 2.

The objective (b1) is to minimize the total cost. Constraints (a2) to (a7) have the same meanings as in electric bus scheduling. Constraints (b3), (b4) and (b5) ensure that the requirement of starting time is satisfied. Constraint (b6) guarantees that the vehicle capacity is not violated.

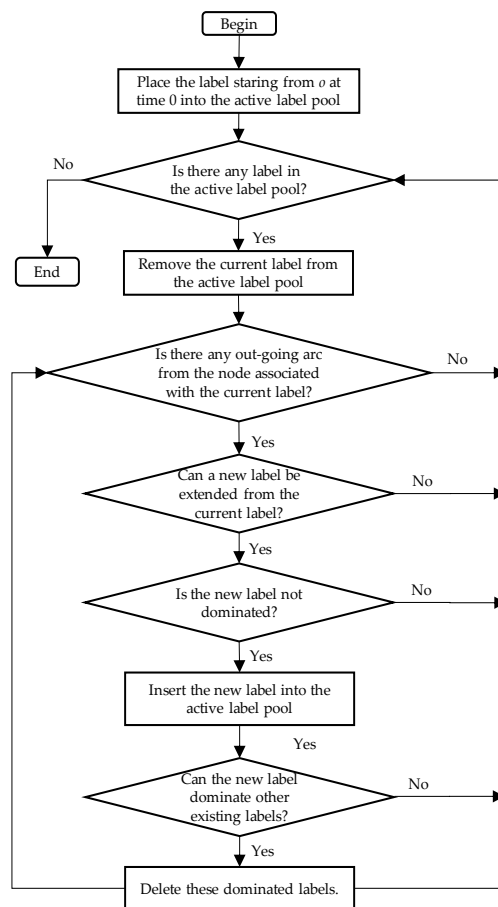
If D is set as a sufficiently large value, the problem becomes the shortest path problem with time windows, which has been proven to be NP-hard in the strong sense [40]. Therefore, the shortest path problem in electric truck routing is NP-hard.

Table 2. The mathematical model of the shortest path problem in electric truck routing with the time windows.

Objective Function and Restrictions	Variable Range	Equation No.
$\min \sum_{(j,i) \in A} \bar{c}_{ji} x_{ji}$		(b1)
st		
(a2) – (a7)		(b2)
$s_j + U_j + t_{ji} \leq s_i + M(1 - x_{ji})$	$\forall (j,i) \in A, i \in S \cup T$	(b3)
$a_i \leq s_i \leq b_i$	$\forall i \in S$	(b4)
$s_0 = 0$		(b5)
$\sum_{(j,i) \in A} v_i x_{ji} \leq V$		(b6)
$x_{ij} \in \{0, 1\}$	$(i,j) \in A$	(b7)

3. The Label-Correcting Algorithm

In this section, we present a label-correcting algorithm to solve the proposed shortest path problems. As discussed in Section 1, preprocessing and K-shortest path enumeration have been successfully used in the resource-constrained shortest path problem without negative cost cycles. Although the underlying network for electric bus scheduling does not contain negative cost cycles, it is difficult to employ these approaches due to the battery renewal. In this paper, we develop a unified label-correcting algorithm to find the optimal elementary shortest paths for both the electric bus scheduling and electric truck routing problems. The details of the label-correcting algorithm (LC) are provided in Figure 1. After the algorithm is finished, an optimal elementary path will be outputted.

**Figure 1.** The flowchart of the label-correcting algorithm.

Two major operations in the label-correcting algorithm are label extension and label dominance. Label extension is critical to ensure the solution feasibility, while label dominance is crucial to algorithm efficiency. Dominated labels can be deleted from the label pool.

3.1. The Label Operations for Electric Bus Scheduling

The label operations are relatively simple for electric bus scheduling due to an acyclic underlying network and fixed service start times. In the extension context (state space), each state can be represented by label (i, g) , where i is the last reached node and g the accumulative traveling distance since the last battery renewal. $\bar{c}(i, g)$ represents the accumulative cost of label (i, g) since leaving from the depot.

The first label in the extension space is $(o, 0)$, which means that the vehicle starts from the depot with distance zero and cost zero. Label (i, g) can be extended to label $(j, \bar{g})$. If $g + d_{ij} \leq D$. Additionally, $\bar{g}$ can be determined by a rule: $\bar{g} = 0$ if $j \in T$, and $\bar{g} = g + d_{ij}$ otherwise. For the label dominance relationship, let label (i, g) and label $(i, \bar{g})$ be two paths to node i . We define label $(i, \bar{g})$ to be dominated by label (i, g) if $c(i, g) \leq c(i, \bar{g})$ and $g \leq \bar{g}$.

3.2. The Label Operations for Electric Truck Routing

The label operations are much more complicated for electric truck routing due to the existence of negative cost cycles and additional constraints on the vehicle capacity and time windows. Each label is now defined as (i, g, s, v, R) , where s represents the earliest time when service can commence at i and v represents the accumulative demands at i . R is a binary resource vector to indicate if every node $i \in S$ is visited [8], aimed at avoiding negative cost cycles: initially, all of the positions in R are set to zero; and if a node is visited, the corresponding position in R is set to one. It is not needed to keep track of visiting information for time-expanded station nodes in T . Due to the fixed service starting time, such nodes cannot be visited more than once in a feasible solution. The cost of label (i, g, s, v, R) is $\bar{c}(i, g, s, v, R)$.

The first label in the extension is $(o, 0, 0, 0, R)$ with $R_i = 0, i \in S$. Now, consider that label (i, g, s, v, R) is extended to label $(j, \bar{g}, \bar{s}, \bar{v}, \bar{R})$. The extension is feasible if: (1) $g + d_{ij} \leq D$; (2) $s + U_i + t_{ij} \leq b_j$; (3) $v + v_j \leq V$; and (4) j has not previously been visited ($R_j = 0$). If feasible, label $(j, \bar{g}, \bar{s}, \bar{v}, \bar{R})$ is determined as follows: $\bar{g} = 0$ if $j \in T$, and $\bar{g} = g + d_{ij}$ otherwise; $\bar{s} = \max\{s + U_i + t_{ij}, a_j\}$; $\bar{v} = v + v_j$; and $\bar{R}_k = R_k$ if $k \neq j$, and $\bar{R}_j = 1$. Then, we discuss the label dominance. Consider that both label (i, g, s, v, R) and label $(i, \bar{g}, \bar{s}, \bar{v}, \bar{R})$ have node i as their last reached node. Label (i, g, s, v, R) dominates label $(i, \bar{g}, \bar{s}, \bar{v}, \bar{R})$ if: (1) $c(i, g, s, v, R) \leq c(i, \bar{g}, \bar{s}, \bar{v}, \bar{R})$; (2) $g \leq \bar{g}$; (3) $s \leq \bar{s}$; (4) $v \leq \bar{v}$; and (5) $R \leq \bar{R}$.

Note that in the label dominance, Condition (5) requires examining the visit information for each customer $i \in S$. Such a strong condition results in a large number of non-dominated labels. In order to reduce the number of non-dominated labels and the computational time, certain studies proposed a state space relaxation technique (see [5,14,41,42] for more information) to relax Condition (5), say to: $|R| \leq |\bar{R}|$ where $|R| = \sum_{i \in S} R_i$ and $|\bar{R}| = \sum_{i \in S} \bar{R}_i$. Due to relaxing the visiting information for each individual customer, satisfying Condition (4) in the label extension is not needed. However, some states may correspond to an infeasible solution that includes negative cycles. In order to obtain a feasible solution, we use a decremental state relaxation technique proposed by Righini and Salani [16] that: (i) initially applies the state relaxation to all of the customer nodes; (ii) calls the dynamic programming algorithm LC; (iii) examines if some nodes have been visited more than once in the solution; (iv) if having the nodes with multiple visits, removes the state relaxation, uses the original extension and dominance rules for these nodes and returns to Step (ii); and (v) stops if no node is visited more than once. Both the label-correcting algorithm and the label correcting with state space relaxation will be tested in the computational experiments.

4. Computational Experiments

In the computational experiments, we tested the performance of the label-correcting algorithm and compared it to one of the best MIP solvers of CPLEX 11.2 [43]. We implemented our algorithm in C++ on a Linux Workstation with a 2-GHz CPU and 16 GB of RAM. When implementing the MIP models in CPLEX, we used the big-M technique to linearize $g_j x_{ji}$ appearing in Constraints (a6) and (a7), and the big-M was set equal to D .

In the following experiments, the battery service time was set as 10 min; the maximum waiting time at a battery station was set as 30 min; and the sampling unit for generating time-expanded station nodes (T) was set as 1 min. We set the maximum running time of CPLEX and label-correcting algorithm as 3600 CPU s. The computational experiments show that in comparison with the state-of-art MIP solver CPLEX, our algorithms spend much less time in finding optimal solutions.

4.1. The Shortest Path in Electric Bus Scheduling with Battery Renewal

For the shortest path problem in electric bus scheduling, we used real-world data from a transit agency in the Bay Area. The transit agency is County Connection, whose data can be obtained from [44]. The data are in the GTFS (General Transit Feed Specification) format which is an open format for packaging scheduled service data. They include stop locations and published weekday and weekend schedules. The schedule includes the starting times at the stop locations. In order to conduct more tests, we divided each weekday schedule into two parts and also generated more instances by randomly shifting the trip starting time within $[-10, 10]$ min from the original value. The ending time was also adjusted by keeping the same trip duration. Table 3 describes the problem instances. Columns 3 provides the number of time-expanded station nodes. We determined the distance between each pair of locations by solving a shortest path problem based on the real-world road geometry data provided by NavTeq. We first determined the nearest intersection to each bus stop. We then solved a one-to-one shortest path problem between each intersection using the Dijkstra algorithm. The travel times were then determined using the obtained distances and average vehicle speed of 25 mph.

Table 3. Description of instances in electric bus scheduling.

Instance	# of Trips	# of Time-Expanded Nodes	Description
S	242	952	published weekend schedules
Sa		951	randomly-generated based S
Sb		943	same as above
Sc		930	same as above
W	947	1132	published weekday schedules
Wa		1136	randomly-generated based W
Wb		1136	same as above
Wc		1130	same as above
W1	463	1073	half of W
W1a		1087	randomly-generated based W1
W1b		1074	same as above
W1c		1084	same as above
W2	484	1132	the other half of W
W2a		1131	randomly-generated based W2
W2b		1124	same as above
W2c		1126	same as above

The operational cost of arc (ji) is defined as: (1) if $j = o$, c_{ji} equals the distance cost plus the fixed vehicle maintenance cost; (2) if $i \in S$, c_{ji} equals the distance cost plus the waiting cost at i ; and (3) if $i \in T$, c_{ji} equals the sum of the distance cost, the waiting cost at i and the battery service cost. The fixed vehicle maintenance cost is set as \$100; the unit waiting cost is set as \$0.1 per minute; and each battery service cost is set as \$5. The unit distance cost is estimated based on the data from Haggis and Beback (2010) [35], which show that it can take about 8 h for a battery charger with 400 V/100 A to fully charge

a battery whose operational distance is approximately 490 km. The average electricity price is set as 10.26 cents per kWh [45]. Finally, the electricity cost per kilometer is 6.61935 cents. The overall cost of arc (ji) is defined as: $\bar{c}_{ji} = c_{ji} - 5 \times (\lambda_i + \lambda_j)$, where λ_i and λ_j are dual values of nodes i and j . In order to conduct the computational experiments, we have to generate dual variables (e.g., see [14]). First, we determined the average arc operational cost by excluding the fixed vehicle cost and set it as C . Then, we generated an integer value uniformly drawn from $[0.7C, 1.3C]$ and set it as the dual variable. In order to conduct more tests, we generated two different dual variables for each instance using different random seeds.

In accordance with real-world operations, we tested two different values of the maximum operational distance: 120 km and 150 km. We considered two different locations for the battery station. In the first case, the battery station is assumed to be located in the depot, and the computational results are shown in Table 4. In Table 4, Columns 1 and 2 provide the problem instance and dual variable ID. Columns 3 to 6 report the results when the maximum operational distance is 120 km. Column 3 provides the optimal objective value. Column 4 reports the CPU seconds of the CPLEX MIP solver. Columns 5 and 6 present the number of non-dominated labels and CPU seconds when the label-correcting algorithm is used. Columns 7 to 10 report the corresponding results when the maximum operational distance is 150 km.

Table 4. Results on scheduling with the battery station in the depot.

Instance	Dual	Max Distance 120 km				Max Distance 150 km			
		Optimal Objective Value	CPLEX	Label Correcting		Optimal Objective Value	CPLEX	Label Correcting	
			CPU (s)	Labels	CPU (s)		CPU (s)	Labels	CPU (s)
S	1	−822,351	117.94	2875	0.72	−824,405	141.92	2897	0.74
	2	−819,804	1167.49	2668	0.72	−829,648	61.65	2739	0.78
Sa	1	−707,440	26.77	1735	0.53	−707,440	27.79	1735	0.55
	2	−705,225	39.38	2065	0.52	−705,225	32.6	2066	0.58
Sb	1	−712,234	45.25	1917	0.54	−712,234	38.13	1919	0.56
	2	−724,879	26.09	2134	0.54	−724,879	29.23	2134	0.6
Sc	1	−751,972	101.99	2305	0.57	−755,469	42.92	2511	0.6
	2	−790,796	33.72	2283	0.56	−790,796	40.46	2302	0.58
W1	1	−1,175,252	2927.46	3239	1.64	−1,175,252	+	3287	1.74
	2	−1,185,467	338.88	3012	1.62	−1,185,467	273.81	2987	1.71
W1a	1	−1,190,403	262.11	3123	1.5	−1,190,403	273.04	3123	1.58
	2	−1,140,363	328.97	3075	1.46	−1,140,363	256.57	3078	1.56
W1b	1	−1,180,257	738.82	3431	1.46	−1,181,390	298.17	3639	1.69
	2	−1,133,505	249.87	2702	1.35	−1,133,505	808.2	2717	1.41
W1c	1	−1,226,202	248.59	3116	1.69	−1,226,202	251.33	3118	1.8
	2	−1,160,532	+	4126	1.53	−1,162,385	1052.89	4513	1.65
W2	1	−1,241,010	1142.48	3060	1.69	−1,241,010	+	3195	1.82
	2	−1,250,571	339.46	3133	1.65	−1,250,571	593.26	3205	1.74
W2a	1	−1,236,314	314.43	3492	1.61	−1,236,314	481.85	3548	1.66
	2	−1,293,550	421.19	2899	1.62	−1,293,550	1513.48	2924	1.68
W2b	1	−1,271,817	3565.16	4166	1.53	−1,286,476	505.13	4906	1.58
	2	−1,228,527	1645.09	3831	1.56	−1,235,273	424.58	4133	1.67
W2c	1	−1,321,741	376.29	3967	1.81	−1,321,741	362.13	3990	1.92
	2	−1,279,684	274.28	3681	1.84	−1,279,684	311.64	3742	1.97
W	1	−1,614,504	*	8333	8.58	−1,628,219	*	9104	9.65
	2	−1,574,276	*	7751	8.13	−1,590,012	*	8420	8.96
Wa	1	−1,465,572	*	7953	7.47	−1,471,389	*	8228	8.21
	2	−1,513,933	*	8779	8.04	−1,518,442	*	9101	9.07
Wb	1	−1,484,716	*	13,329	8.24	−1,495,817	*	16,182	9.42
	2	−1,469,219	*	8644	7.99	−1,502,780	*	10,332	8.74
Wc	1	−1,490,073	*	8952	8.04	−1,498,017	*	10,500	9.36
	2	−1,518,417	*	12,600	8.72	−1,526,627	*	14,393	9.74

* CPLEX was terminated due to being out of memory; +: no an optimal solution was found within 1 CPU hour.

In the second case, the station is assumed to be located in a bus stop that roughly centers on the overall coverage area. Figures 2 and 3 show the comparison of the computational times consumed

by CPLEX and label correcting for solving the problem instances for the same scenarios as in Table 4 under this case.

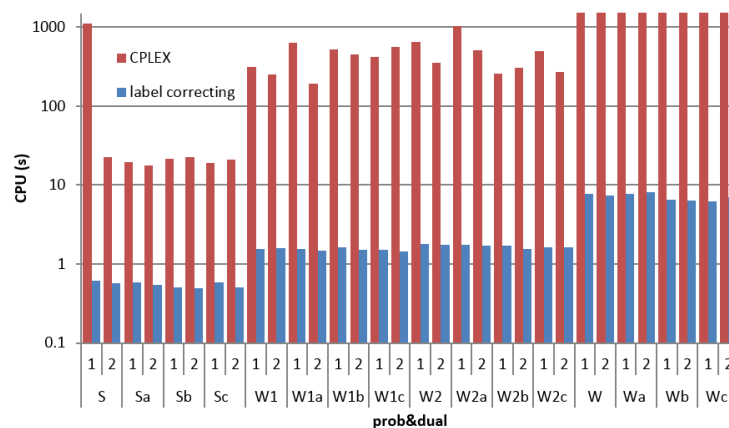


Figure 2. The computational times by CPLEX and label correcting on scheduling with a maximum distance of 120 km and the battery station centering on the service area.

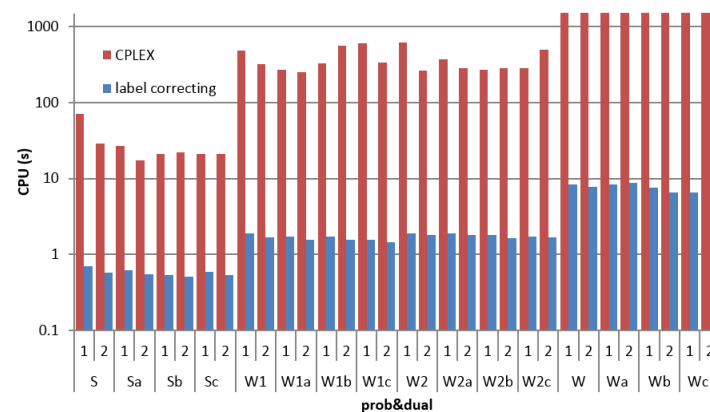


Figure 3. The computational times by CPLEX and label correcting on scheduling with a maximum distance of 150 km and the battery station centering on the service area.

Table 4 and Figures 2 and 3 show that the label-correcting algorithm finds the optimal solution very quickly for the instances with less than 500 trips (instances S, W1, W2 and ones that are randomly generated based on them); the computational time is less than two CPU seconds. For the large instances with 947 trips (instances W, Wa, Wb and Wc), the label-correcting algorithm spends less than 10 CPU seconds to obtain optimal solutions. The label-correcting algorithm requires much smaller computational times than CPLEX for all of the instances tested. In particular, for large instances W, Wa, Wb and Wc, CPLEX was terminated due to insufficient memory (see Columns 4 and 8 in Table 4, and the last eight red bars in Figures 2 and 3). The computational time of CPLEX is larger if the maximum operational distance is smaller (for example, see Columns 4 and 8 in Table 4). In contrast, the label-correcting algorithm spends less time to obtain optimal solutions in such situations, which can be confirmed by the lesser number of non-dominated labels (see Columns 5 and 9 in Table 4). Such phenomena may be explained as follows. If the maximum operational distance is smaller, the linear programming model is more tightly constrained and is certainly more difficult for CPLEX to solve. However, in the label-correcting algorithm, the existing labels have to extend to time-expanded station nodes in T more frequently. While at time-expanded station nodes, g of all the labels is zero, and only cost $\bar{c}(i, g)$ needs to be compared in the label dominance. Therefore, the label dominance is stronger on average at time-expanded station nodes, resulting in lesser number of non-dominated labels in total. The overall computational time is then reduced.

4.2. The Shortest Path in Electric Truck Routing with Battery Renewal

As discussed in Section 2.2, the current battery technique allows a truck to run about 120 to 140 km [39,40]. We tested two different values of the maximum operational distance: one was 130 km, and the other one was shorter, 100 km. In order to test the impact of the maximum operational distance in the context of real-world roads, we calculated the travel distance between two locations based on the road geometry data provided by NavTeq. We did not use the standard benchmark instances by [46]. The South Bay Area in California was selected as our case study. First, we determined $|S|$ locations by randomly generating latitude and longitude in the South Bay Area. Then, we conducted the point-to-curve map matching to project the random location to the link in the road geometry network with the shortest perpendicular distance [47]. The point on the projected link was selected as the customer location. Finally, we solved a one-to-one shortest distance problem between each customer location using the Dijkstra algorithm based on the road geometry network. The travel times were then determined using the obtained distances and the average vehicle speed of 40 mph.

The demand at each location was randomly selected from 10, 20 and 30. The vehicle capacity was set as 400. The earliest service start time (a_i) at node $i \in S$ was uniformly drawn from 8 a.m. to 6 p.m. As is well known, the width of time windows has a great impact on the computational time. We tested three scenarios in terms of the width range of time windows: [5, 20] min, [5, 40] min and [5, 60] min. In each scenario, the width of time windows at each node $i \in S$ was uniformly drawn from the given range. Table 5 presents the summary of generated problem instances. The operational cost of arc ji is defined as the distance cost. If i is a time-expanded station node (i.e., $i \in T$), an additional \$5 is added for battery service cost. The unit distance cost was estimated based on the data from Ford Transit Connect [48], which shows that it can take about 8 h for a 6.6-kilowatt charger to fully charge a battery whose operational distance is approximately 120 km. The average electricity price was set as 10.26 cents per kWh [43]. Finally, the electricity cost per kilometer is 4.207677 cents. The overall cost of arc (ji) is defined as: $\bar{c}_{ji} = c_{ji} - 5 \times (\lambda_i + \lambda_j)$, where λ_i are λ_j generated in a similar way for electric bus scheduling.

Table 5. Description of the instances in electric truck routing.

Instance	# of Customers	# of Time-Expanded Nodes			Instance	# of Customers	# of Time-Expanded Nodes	
		Time Windows					Time Windows	
		[5, 20] min	[5, 40] min	[5, 60] min			[5, 20] min	[5, 40] min
50a	50	620	665	685	150a	150	662	695
50b	50	585	666	687	150b	150	636	677
50c	50	623	661	681	150c	150	652	716
50d	50	597	641	665	150d	150	638	696
50e	50	547	644	653	150e	150	665	686
100a	100	644	656	662	200a	200	667	678
100b	100	634	639	663	200b	200	664	680
100c	100	655	693	696	200c	200	633	680
100d	100	645	680	700	200d	200	645	660
100e	100	658	683	697	200e	200	656	694

We first briefly show the effectiveness of the state relaxation in Figure 4 with CPU seconds from eight problem instances. Figure 4 clearly shows that the state relaxation effectively reduces the computational time. From now on, we will only use the label correcting with state relaxation for further discussions. Table 6 presents the computational results for the width of time window [5, 20]. Columns 3 to 7 report the results when the maximum operational distance is 100 km. The column definitions are similar to the ones in Table 4 with the exception of Column 4, which presents the optimality gap of CPLEX, defined as $|(objective\ value - optimal\ value) / objective\ value|$, among which, the objective value is obtained from the MIP solver of CPLEX; the optimal value is from the label-correcting algorithm; and the absolute value operator is used since both values are negative. Columns 8 to 12 report the corresponding results when the maximum operational distance is 130 km.

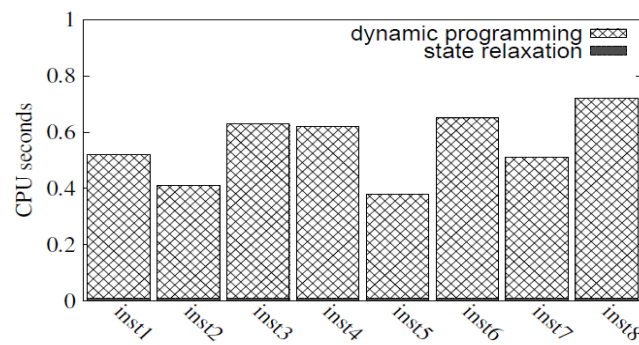


Figure 4. The performance of the dynamic programming with state relaxation.

Table 6. Results on electric truck routing with the time window width in [5, 20] min.

Instance	Dual	Max Distance 100 km					Max Distance 130 km				
		Optimal Objective Value	CPLEX		State Relaxation		Optimal Objective Value	CPLEX		State Relaxation	
			Optimal Gap	CPU (s)	Labels	CPU (s)		Optimal Gap	CPU (s)	Labels	CPU (s)
50a	1	−6220	0.00	0.63	1744	0.02	−6220	0.00	0.70	1780	0.02
	2	−5314	0.00	0.63	1491	0.02	−5314	0.00	0.61	1539	0.02
50b	1	−8742	0.00	0.57	2896	0.01	−8742	0.00	0.57	2904	0.01
	2	−8320	0.00	0.55	3204	0.02	−8320	0.00	0.57	3208	0.02
50c	1	−7195	0.00	0.50	2657	0.01	−7195	0.00	0.50	2771	0.02
	2	−7487	0.00	0.53	2629	0.01	−7487	0.00	0.49	2824	0.01
50d	1	−12,234	0.00	0.54	3412	0.02	−12,234	0.00	0.54	3846	0.02
	2	−11,807	0.00	0.59	4405	0.02	−11,807	0.00	0.56	4841	0.02
50e	1	−7986	0.00	0.47	2163	0.01	−7986	0.00	0.49	2404	0.01
	2	−7962	0.00	0.48	2287	0.01	−7962	0.00	0.45	2536	0.01
100a	1	−11,773	0.00	1.66	5088	0.04	−11,773	0.00	1.45	5483	0.04
	2	−12,385	0.00	1.37	5364	0.05	−12,385	0.00	1.35	5754	0.04
100b	1	−12,292	0.00	3.84	9008	0.09	−13,195	0.00	1.55	9588	0.1
	2	−10,774	0.00	5.41	9615	0.10	−11,494	0.00	1.59	10,277	0.11
100c	1	−11,655	0.00	1.27	6211	0.05	−11,655	0.00	1.23	6446	0.06
	2	−9025	0.00	1.33	5253	0.04	−9025	0.00	1.38	5539	0.04
100d	1	−13,405	0.00	1.70	9543	0.08	−13,405	0.00	1.55	9816	0.09
	2	−12,576	0.00	1.53	10,192	0.10	−12,576	0.00	1.67	10,391	0.1
100e	1	−13,473	0.00	1.61	10,546	0.10	−13,473	0.00	1.50	11,208	0.11
	2	−14,937	0.00	1.57	8134	0.08	−14,937	0.00	1.53	8628	0.08
150a	1	−14,980	0.00	4.99	20,942	0.84	−16,159	0.00	3.48	21,548	0.84
	2	−16,327	0.00	6.80	18,039	0.75	−16,883	0.00	4.66	18,726	0.75
150b	1	−15,493	0.00	4.56	12,588	0.14	−15,493	0.00	4.54	12,913	0.15
	2	−14,026	0.00	6.85	9966	0.13	−14,026	0.00	5.86	10,167	0.14
150c	1	−17,062	0.00	3.18	14,051	0.62	−17,062	0.00	5.24	14,118	0.65
	2	−16,292	0.00	4.13	18,626	1.50	−16,292	0.00	4.32	18,738	1.58
150d	1	−18,158	0.00	10.70	29,828	3.36	−19,656	0.00	4.61	31,441	2.34
	2	−17,744	0.00	12.90	19,941	1.48	−18,587	0.00	5.99	26,233	1.87
150e	1	−17,925	0.00	43.71	23,911	0.42	−19,214	0.00	11.70	26,744	0.44
	2	−19,397	0.00	24.29	23,281	0.38	−21,270	0.00	6.96	25,491	0.39
200a	1	−17,718	0.00	15.93	25,200	2.40	−19,265	0.00	13.09	24,390	0.53
	2	−18,891	0.00	24.28	30,553	2.68	−18,891	0.00	10.14	26,201	2.37
200b	1	−21,300	0.00	13.07	26,969	0.53	−22,216	0.00	7.18	27,031	0.54
	2	−22,300	0.00	12.21	20,303	0.36	−23,112	0.00	6.95	20,396	0.38
200c	1	−18,336	0.00	14.00	23,512	1.29	−19,260	0.00	6.19	25,316	1.43
	2	−17,194	0.00	14.90	25,744	2.46	−18,212	0.00	6.54	26,967	2.46
200d	1	−19,536	0.00	27.92	46,037	4.80	−20,742	0.00	67.16	48,768	4.93
	2	−20,315	0.00	92.37	46,088	2.80	−20,607	0.00	58.97	46,342	5.51
200e	1	−20,541	0.00	14.92	30,201	0.62	−20,541	0.00	18.21	30,808	0.65
	2	−21,171	0.00	36.28	20,917	1.00	−22,303	0.00	21.47	21,413	0.98

Table 6 shows that if the width of time windows is randomly drawn from $[5, 20]$ min, the label correcting with state relaxation finds an optimal solution within 0.2 s for the instances with 50 and 100 customers. If the number of customers is 150 or 200, the CPU seconds are in the range of $[0.14, 5.51]$ s. The label-correcting algorithm requires much smaller computational times than CPLEX for all of the instances with the width of time windows in $[5, 20]$. Figure 5 shows the comparison of computational times by CPLEX and label correcting on electric truck routing with the time window width in $[5, 40]$ min and with a maximum distance of 100 km. Figure 6 shows the corresponding comparison with a maximum distance of 130 km. Figure 7a,b shows the comparisons of computational times with the time window width in $[5, 60]$ min, with a maximum distance of 100 km and 130 km, separately. We can find when the width of time windows is in the range $[5, 40]$, the label-correcting algorithm still finds an optimal solution for the instances with no more than 100 customers within 3 min (see Figure 5). When the width of time windows is in $[5, 60]$, the computational time is less than 5 min for the instances with 50 customers (see Figure 6). However, if both the number of customers and the width of time windows increase, the computational time of the label correcting often becomes substantially longer, for example instances 150a to 150e with time windows in $[5, 40]$ in Figures 5 and 6; and several instances cannot be solved by the label-correcting algorithm within one CPU hour. In comparison with the MIP solver of CPLEX, the label-correcting algorithm performs much better for most instances tested. In addition, similar to the cases in electric bus scheduling, if the maximum operational distance is smaller, the computational time of CPLEX becomes longer, while on average, the label-correcting algorithm spends less time to obtain an optimal solution. In addition, it is worthy to mention that we generated relatively large dual variables, which result in more negative cost cycles (see the objective values in Columns 3 and 8 in Table 6) and longer computational time (see Columns 5, 7, 10 and 11 in Table 6, Figures 5–7).

Finally, the computational experiments show that the shortest path problem in electric truck routing is much more difficult to solve than the shortest path problem in electric bus scheduling. It takes the label-correcting algorithm less than 9 s to find the shortest paths for bus scheduling problems with 947 trips, while the label correcting may spend a much longer time to find the shortest paths for a routing instance with 100 customers and relatively wide time windows. The long computational time in electric truck routing is mainly caused by the existence of negative cost cycles in the underlying network, which leads to much more complicated label extension and dominance. The comparison of the number of non-dominated labels in electric bus scheduling and electric truck routing supports such an observation.

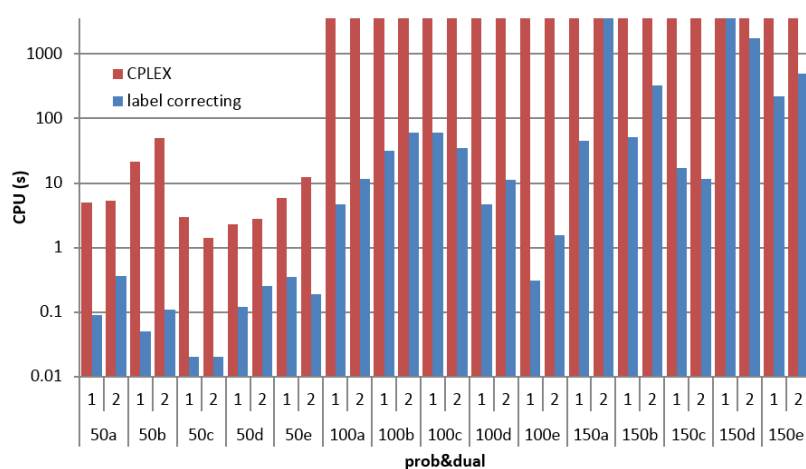


Figure 5. The computational times by CPLEX and label correcting on electric truck routing with the time window width in $[5, 40]$ min and a maximum distance of 100 km.

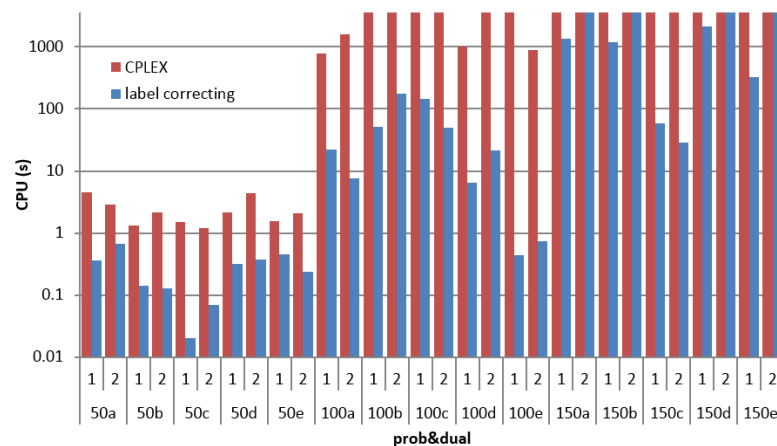


Figure 6. The computational times by CPLEX and label correcting on electric truck routing with the time window width in $[5, 40]$ min and a maximum distance of 130 km.

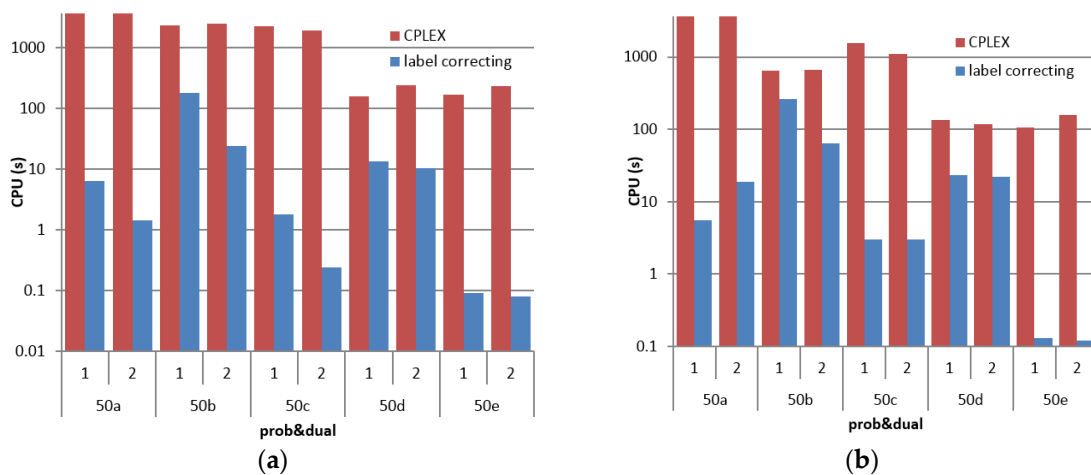


Figure 7. The computational times by CPLEX and label correcting on electric truck routing with the time window width in $[5, 60]$ min. (a) with a maximum distance of 100 km; (b) with a maximum distance of 130 km.

5. Conclusions and Future Research

In this paper, we study the shortest path problems in emerging electric vehicle dispatching, where a fully-charged battery allows a vehicle to run a limited operational distance and the battery near depletion has to be charged or exchanged with a fully-charged one at a battery management facility. The battery renewal makes the problem different from the resource-constrained shortest path problems in the literature. We investigate the shortest paths for two important applications: one is electric transit bus scheduling, and the other is electric truck routing with time windows. We develop a label-correcting algorithm to solve these shortest path problems. Since the underlying network of electric truck routing may contain negative cost cycles, the state space relaxation technique is incorporated into the label correcting to reduce the computational time.

In the computational experiments, the travel time and distance are determined based on the real-world road geometry data. Other types of data are obtained from the real-world or randomly generated. The label-correcting algorithm spends less than 9 s to find the shortest paths for electric bus scheduling problems with 947 trips. Due to the existence of negative cost cycles, the shortest path problem in electric truck routing is much more difficult to solve than in the electric bus scheduling. The label correcting with state space relaxation quickly obtains optimal solutions for the instances with

200 customers and the width of time windows in [5, 20] min. However, if both the number of customers and the width of time windows increase, the computational time often becomes substantially longer. We compare the label-correcting algorithm developed with the CPLEX MIP solver. The computational results show that the label-correcting algorithm performs much better than the CPLEX for the most instances tested, since the former is specially designed to solve the shortest path problem in electric vehicle dispatching. If the maximum operational distance is smaller, CPLEX spends a longer time for the optimal solution, while the label-correcting algorithm, on average, requires a shorter time.

The shortest path problem is a critical component in column generation algorithms since the sub-problem that is generally a shortest path problem needs to be repeatedly solved. Column generation has been successfully used in many studies to solve complicated integer programming problems [6,7]. Therefore, it is very important to develop an efficient algorithm for the shortest path problem with battery renewal. We expect that our study can play an important role in electric vehicle optimization problems.

Future work can include the development of new algorithms to reduce the computational time for instances with wide time windows. The shortest path models and algorithms can be incorporated into a column generation framework to solve electric vehicle dispatching. Investigating the location problem of battery management facilities is another topic.

Acknowledgments: This work is partially supported by the grants from the National Natural Science Foundation of China (#71201055, #71571067) and Beijing Municipal Philosophy and Social Science Planning (12JGC080).

Author Contributions: Both authors equally contributed to designing the study, analyzing the data and writing the paper.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Environmentally Friendly Options and UPS Services. Available online: http://www.ups.com/content/us/en/bussol/browse/article/small-business-sustainability.html?srch_pos=2&srch_phr=electric+delivery+trucks (accessed on 10 December 2015).
2. United Nations Environment Programme. *UNEP Environmental Assessment Expo 2010 Shanghai, China*; Tech. Rep. DCP/1209/NA. 2009. Available online: <http://www.sepb.gov.cn/platform/UserFiles/File/2009-08-18-16-53-46-884+0800-570148450609847911.pdf> (accessed on 23 June 2016).
3. Beijing Electric Vehicle Promotion Application Action Plan (2014–2017). Available online: <http://www.zhengwu.beijing.gov.cn/ghxx/qtgh/t1359600.htm> (accessed on 12 December 2015).
4. Desaulniers, G.; Lavigne, J.; Soumis, F. Multi-depot vehicle scheduling problems with time windows and waiting costs. *Eur. J. Oper. Res.* **1998**, *3*, 479–494. [[CrossRef](#)]
5. Desrocher, M.; Desrosier, J.; Solomon, M.M. A new optimization algorithm for the vehicle-routing problem with time windows. *Oper. Res.* **1992**, *2*, 342–354. [[CrossRef](#)]
6. Desrochers, M.; Soumis, F. A column generation approach to the urban transit crew scheduling problem. *Transp. Sci.* **1989**, *1*, 1–13. [[CrossRef](#)]
7. Muter, I.; Birbil, S.I.; Bulbul, K.; Sahin, G.; Yenigun, H.; Tas, D.; Tuzun, D. Solving a robust airline crew pairing problem with column generation. *Comput. Oper. Res.* **2013**, *3*, 815–830. [[CrossRef](#)]
8. Beasley, J.E.; Christo, N. An algorithm for the resource-constrained shortest-path problem. *Networks* **1989**, *3*, 379–394. [[CrossRef](#)]
9. Desrochers, M.; Soumis, F. A generalized permanent labeling algorithm for the shortest path problem with time windows. *INFOR* **2014**, *3*, 191–212.
10. Dumitrescu, I.; Boland, N. Improved preprocessing, labeling and scaling algorithms for the weight-constrained shortest path problem. *Networks* **2003**, *3*, 135–153. [[CrossRef](#)]
11. Handler, G.Y.; Zang, I. A dual algorithm for the constrained shortest path problem. *Networks* **2006**, *4*, 293–309. [[CrossRef](#)]
12. Santos, L.; Coutinho-Rodrigues, J.; Current, J.R. An improved solution algorithm for the constrained shortest path problem. *Transp. Res. Part B Methodol.* **2006**, *7*, 756–771. [[CrossRef](#)]

13. Xing, T.; Zhou, X. Reformulation and solution algorithms for absolute and percentile robust shortest path problems. *IEEE Trans. Intell. Transp. Syst.* **2013**, *2*, 943–954. [[CrossRef](#)]
14. Feillet, D.; Dejax, P.; Gendreau, M.; Gueguen, C. An exact algorithm for the elementary shortest path problem with resource constraints: Application to some vehicle routing problems. *Networks* **2004**, *3*, 216–229. [[CrossRef](#)]
15. Righini, G.; Salani, M. Symmetry helps: Bounded bi-directional dynamic programming for the elementary shortest path problem with resource constraints. *Discret. Optim.* **2006**, *3*, 255–273. [[CrossRef](#)]
16. Righini, G.; Salani, M. New dynamic programming algorithms for the resource constrained elementary shortest path problem. *Networks* **2008**, *3*, 155–170. [[CrossRef](#)]
17. Houck, D.J.; Picard, J.C.; Queyranne, M.; Vemuganti, R.R. The travelling salesman problem as a constrained shortest path problem: Theory and computational experience. *Oper. Res.* **1980**, *2*, 93–109.
18. Irnich, S.; Villeneuve, D. The shortest-path problem with resource constraints and k-cycle elimination for $k > 3$. *INFORMS J. Comput.* **2006**, *3*, 391–406. [[CrossRef](#)]
19. Bode, C.; Irnich, S. The shortest-path problem with resource constraints with (k, 2)-loop elimination and its application to the capacitated arc-routing problem. *Eur. J. Oper. Res.* **2014**, *2*, 415–426. [[CrossRef](#)]
20. Garey, M.R.; Johnson, D.S. *Computers and Intractability: A Guide to the Theory of NP-Completeness*, 1st ed.; W.H. Freeman: New York, NY, USA, 1979.
21. Wang, Y.W.; Lin, C.C. Locating road-vehicle refueling stations. *Transp. Res. Part E Logist. Transp. Rev.* **2009**, *5*, 821–829. [[CrossRef](#)]
22. Yildiz, B.; Arslan, O.; Karasan, O.E. A branch and price approach for routing and refueling station location model. *Eur. J. Oper. Res.* **2015**, *3*, 815–826.
23. Huang, S.; He, L.; Gu, Y.; Wood, K.; Benjaafar, S. Design of a mobile charging service for electric vehicles in an urban environment. *IEEE Trans. Intell. Transp. Syst.* **2014**, *2*, 787–798. [[CrossRef](#)]
24. Kang, Q.; Wang, J.; Zhou, M.; Ammari, A.C. Centralized charging strategy and scheduling algorithm for electric vehicles under a battery swapping scenario. *IEEE Trans. Intell. Transp. Syst.* **2015**, *1*, 1–11. [[CrossRef](#)]
25. Roberti, R.; Wen, M. The electric traveling salesman problem with time windows. *Transp. Res. Part E Logist. Transp. Rev.* **2016**, *89*, 32–52. [[CrossRef](#)]
26. Bruglieri, M.; Pezzella, F.; Pisacane, O.; Suraci, S. A variable neighborhood search branching for the electric vehicle routing problem with time windows. *Electron. Notes Discret. Math.* **2015**, *47*, 221–228. [[CrossRef](#)]
27. Yang, J.; Sun, H. Battery swap station location-routing problem with capacitated electric vehicles. *Comput. Oper. Res.* **2015**, *55*, 217–232. [[CrossRef](#)]
28. Yagcitek, B.; Uzunoglu, M. A double-layer smart charging strategy of electric vehicles taking routing and charge scheduling into account. *Appl. Energy* **2016**, *167*, 407–419. [[CrossRef](#)]
29. Elkins, J.; Hemby, J.; Rao, V.; Szykman, J.; Fitz-Simons, T.; Doll, D.; Mintz, D.; Rush, A.; Frank, N.; Schmidt, M.; et al. *National Air Quality and Emission Trends Report Special Studies Edition*; U.S. Environmental Protection Agency: North Carolina, CA, USA, 2003.
30. O'Brien, W.A. Electric vehicles (EVs): A look behind the scenes. *IEEE Aerosp. Electron. Syst. Mag.* **1993**, *5*, 38–41. [[CrossRef](#)]
31. Argonne National Laboratory. Electric Buses Energize Downtown Chattanooga. Available online: http://www.afdc.energy.gov/afdc/pdfs/chatt_cs.pdf (accessed on 12 December 2015).
32. Beijing Readies Electric Buses for Summer Olympics. Available online: <http://www.evworld.com/news.cfm?newsid=18730> (accessed on 13 December 2015).
33. Namsan Electric Bus Cruises into International Limelight. Available online: <http://www.cnn.go.com/seoul/life/namsan-electric-bus-cruises-limelight-262862> (accessed on 13 December 2015).
34. Quick-charge Electric Bus Rolls into L.A. County. Available online: <http://www.wired.com/autopia/2010/09/proterra-ecoride-foothill-transit> (accessed on 13 December 2015).
35. Transi. Bus Technology Feasibility Study. Available online: http://www.advancedenergy.org/transportation/additional_efforts/TransitBusFeasibilityStudy_Final.pdf (accessed on 20 December 2015).
36. Final Environmental Review of the 2010 World Exposition. Available online: <http://www.unep.org/newscentre/Default.aspx?DocumentID=2659&ArticleID=8964&l=en#sthash.zr7SV1wK.dpuf> (accessed on 15 November 2015).
37. Shafia, M.A.; Aghaee, M.P.; Sadjadi, S.J.; Jamili, A. Robust train timetabling problem: Mathematical model and branch and bound algorithm. *IEEE Trans. Intell. Transp. Syst.* **2012**, *1*, 307–317. [[CrossRef](#)]

38. Li, J.Q. Transit bus scheduling with limited energy. *Transp. Sci.* **2013**, *4*, 521–539. [[CrossRef](#)]
39. UPS' Brown Trucks Get Greener with Electric Power. Available online: <http://www.sfgate.com/cgi-bin/article.cgi?f=/c/a/2011/08/25/BU711KRF10.DTL> (accessed on 12 December 2015).
40. Ford Transit Connect Electric Commercial van Helps Customers Go Completely Gas-Free. Available online: http://www.media.ford.com/article_display.cfm?article_id=31991 (accessed on 12 December 2015).
41. Boland, N.; Dethridge, J.; Dumitrescu, I. Accelerated label setting algorithms for the elementary resource-constrained shortest path problem. *Oper. Res. Lett.* **2004**, *1*, 58–68. [[CrossRef](#)]
42. Christo, N.; Mingozzi, A.; Toth, P. State-space relaxation procedures for the computation of bounds to routing problems. *Networks* **2006**, *2*, 145–164.
43. IBM Cplex. Available online: <https://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/> (accessed on 9 June 2016).
44. GTFS data. Available online: <http://www.gtfs-data-exchange.com/agency/county-connection> (accessed on 12 December 2015).
45. Energy Information Administration. (2011, Dec). Prices and Factors Affecting Prices. Available online: http://www.eia.gov/energyexplained/index.cfm?page=electricity_factors_affecting_prices (accessed on 12 December 2015).
46. Solomon, M.M. Algorithms for the vehicle-routing and scheduling problems with time window constraints. *Oper. Res.* **1987**, *2*, 254–265. [[CrossRef](#)]
47. Li, J.Q. Match bus stops to a digital road network by the shortest path model. *Transp. Res. Part C Emerg. Technol.* **2012**, *2*, 119–131. [[CrossRef](#)]
48. Ford Working with Best Buy to Offer Focus Electric Charging Station Sales, Installation and Support. Available online: http://www.media.ford.com/article_display.cfm?article_id=33772 (accessed on 12 December 2015).



© 2016 by the authors; licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC-BY) license (<http://creativecommons.org/licenses/by/4.0/>).