http://pubsonline.informs.org/journal/trsc/

TRANSPORTATION SCIENCE
Vol. 53, No. 4, July–August 2019, pp. 1107–1125
ISSN 0041-1655 (print), ISSN 1526-5447 (online)

A Branch-and-Cut Algorithm for the Alternative Fuel Refueling Station Location Problem with Routing

Okan Arslan,[a] Oya Ekin Karaşan,[b] A. Ridha Mahjoub,[c] Hande Yaman[b]

[a] HEC Montréal and CIRRELT, Montréal, Quebec H3T 2A7 Canada; [b] Department of Industrial Engineering, Bilkent University, Bilkent, 06800 Ankara, Turkey; [c] Université Paris Dauphine, PSL Research University, CNRS [7243], LAMSADE, 75016 Paris, Île-de-France, France

Contact: okan.arslan@hec.ca, https://orcid.org/0000-0002-7862-3449 (OA); karasan@bilkent.edu.tr (OEK); ridha.mahjoub@lamsade.dauphine.fr (ARM); hyaman@bilkent.edu.tr, https://orcid.org/0000-0002-3392-1127 (HY)

Received: June 20, 2017
Revised: May 3, 2018
Accepted: August 9, 2018
Published Online in Articles in Advance: June 13, 2019

https://doi.org/10.1287/trsc.2018.0869

Abstract. Because of the limited range of alternative fuel vehicles (AFVs) and the sparsity of the available alternative refueling stations (AFSs), AFV drivers cooperatively deviate from their paths to refuel. This deviation is bounded by the drivers' tolerance. Taking this behavior into account, the refueling station location problem with routing (RSLP-R) is defined as maximizing the AFV flow that can be accommodated in a road network by locating a given number of AFSs while respecting the range limitation of the vehicles and the deviation tolerance of the drivers. In this study, we develop a natural model for the RSLP-R based on the notion of length-bounded cuts, analyze the polyhedral properties of this model, and develop a branch-and-cut algorithm as an exact solution approach. Extensive computational experiments show that the algorithm significantly improves the solution times with respect to previously developed exact solution methods and extends the size of the instances solved to optimality. Using our methodology, we investigate the tradeoffs between covered vehicle flow and deviation tolerance of the drivers and present insights on deviation characteristics of drivers in a case study in California.

Funding: This work was partially supported by the Scientific and Technological Research Council of Turkey (TUBITAK) [Grant 214M211]. A. R. Mahjoub's research was partially supported by TUBITAK [Programme 2221].
Supplemental Material: The online appendix is available at https://doi.org/10.1287/trsc.2018.0869.

Keywords: alternative fuel vehicles • refueling station • location • routing • branch-and-cut • length-bounded cut

1. Introduction

The purpose of this paper is to develop a branch-and-cut (B&C) algorithm for the refueling station location problem with routing (RSLP-R). In a transportation network with alternative fuel vehicles (AFVs) traveling between their origin–destination (OD) pairs, the RSLP-R is defined as locating a given number of alternative fuel stations (AFSs) at the nodes of the network such that the total vehicle flow traveling without running out of fuel on paths whose lengths are bounded by the tolerance of the drivers is maximized. This problem is closely related with the efforts to curtail the negative impacts of fossil fuels on the environment. Dependence on fossil fuels contributes significantly to many of the environmental problems we face today, such as air pollution, globally increasing temperatures, and climate change. Because of their limited supply levels, reserves are destined to deplete. Historically, the transportation sector has been one of the major consumers of fossil fuels. However, there has been a recent surge for alternative forms of fuels to be used in transportation. These include hydrogen, biodiesel, electricity, ethanol, compressed natural gas, and liquefied natural gas (Energy Information Administration 2017a). In 2016, in the United States, 24.79 million passenger cars and light trucks using alternative energy sources were in use. This number accounts to 10.25% of the total number of the same vehicle types in the country, and it is projected to surpass 20% by 2030 (Energy Information Administration 2017b). The AFV usage in Europe is also following a similar trend. The Renewable Energy Directive (European Union 2009) set a 10% target for use of energy from renewable sources in transportation by 2020, a considerable increase from the 4.7% target of 2013 (European Commission 2013).

Because of the rather limited range of AFVs, availability of refueling stations in intercity transportation is a serious barrier to the proliferation of these vehicles. In this regard, location of AFSs has been the topic of several recent articles. There have been two main streams of research, a set-covering location perspective (Wang and Lin 2009, 2013; Wang and Wang 2010) and a maximum covering location perspective. The latter attracted more attention because covering all the demand requires location of numerous facilities, which seems implausible in the initial setup period of the AFS infrastructure. To this end, Kuby and Lim (2005) presented the *flow refueling location problem*, which is defined as maximizing the AFV flow that can be accommodated in a road network by locating p number of AFSs while respecting the range limitation of the vehicles. The demand is defined as the AFVs flowing on fixed paths between their OD pairs. "Flow refueling location problem" and "refueling station location problem" are used interchangeably in the literature to refer to the same problem (MirHassani and Ebrazi 2013, Yıldız et al. 2016). In this paper, we prefer to use the latter. Kuby and Lim (2005) proposed a maximal covering location model (MCLM; Church and ReVelle 1974), in which the coverage of an AFV flow requires location of possibly multiple facilities on its path. The formulation requires a priori generation of all node combinations that enable complete traversal of a path, which is computationally costly. For this reason, heuristic algorithms have been proposed (Lim and Kuby 2010).

Since then, considerable effort has been spent for improving the solution times and the size of the solvable RSLP instances. Two different exact solution approaches stand out in the literature. The first one, presented by MirHassani and Ebrazi (2013), builds on the idea that an AFV travels on the shortest path between two consecutive refueling stops. Given a fixed path, not necessarily the shortest, the authors first build a graph in which any path with a refueling station at all of its intermediate nodes corresponds to a trip in the original path that can be traveled without running out of fuel. Because of this key observation, no preprocessing is necessary to generate node combinations as in Kuby and Lim (2005), and the solution process is significantly accelerated. In the following section, we further elaborate on this transformation that has also been used by Chen et al. (2010) in the context of the regenerator placement problem. The second work, by Capar et al. (2013), refines the modeling logic using the fact that, to cover a path, every arc on this path needs to be traversed using one of the open stations. This new model is similar to the MCLM but does not require explicit generation of feasible node combinations to cover a path. Therefore, both the preprocessing and the solution times improve extensively. Further improvements on the solution times and the size of the solvable instances are obtained by applying Benders decomposition on this model (Arslan and Karaşan 2016).

In this study, we focus our attention on the RSLP-R introduced by Kim and Kuby (2012). In this version of the problem, the path between an OD pair is not fixed and an AFV flow is considered as refueled if there exists a feasible path whose length is within a certain bound. This bound is the tolerance of the driver to deviate from the shortest path. Kim and Kuby (2012) present a mixed integer programming formulation, Kim and Kuby (2013) propose a heuristic approach, and Yıldız et al. (2016) present a branch-and-price (B&P) algorithm to solve this problem. The computational gains of the B&P framework with respect to the original formulation by Kim and Kuby (2012) turned out to be significant.

Another closely related problem is the regenerator placement problem (Yetginer and Karasan 2003, Chen et al. 2010) defined in the context of telecommunications to locate the regenerators that extend the optical reach. From the functional perspective of extending the reach, the AFSs are similar to the regenerators. One major difference is related to the way demand is modeled. The drivers in our application have a distance tolerance on the deviation from the shortest path, which is not present in signal routing. This constitutes an additional challenge to our design problem. Computational findings in the regenerator placement problem show that B&C approaches outperform other exact solution methods, especially in large-scale instances (Rahman et al. 2015, Yıldız and Karaşan 2015, Li and Aneja 2017). We also observe similar results.

In this paper, we propose a natural formulation for the RSLP-R and provide a polyhedral study of the convex hull of integer feasible solutions. We then devise a B&C algorithm as an exact solution technique to solve the problem. The constraints of our formulation, which are exponential in number, are added using a cutting-plane framework. The separation problem boils down to finding a length-bounded node cut in a transformed network. For separating integer solutions, we provide a polynomial time-exact separation algorithm to generate cuts. For fractional solutions, we provide a formulation to generate violated cuts. We also make use of a heuristic algorithm to separate fractional solutions. With our approach, we can solve real-world problem instances, which could only be solved previously by a B&P algorithm in a three-hour time frame, within two minutes. We also further extend the size of the instances that can be solved to optimality. Our approach also addresses multiple vehicle types and possible nonsimple path occurrences in the routes. Using our methodology, we investigate the tradeoffs between covered vehicle flow and deviation tolerance of the drivers and present insights on deviation characteristics of drivers in a case study in California.

In the following, we introduce this new formulation, investigate its polyhedral properties, and present our B&C algorithm along with the results of our computational experiments.

2. Definitions, Formulations, and Polyhedral Analysis

Consider a road network represented by a weighted directed graph $G = (N, A)$ with node set $N = \{1, \ldots, n\}$ and arc set A. Let δ_{ij} be the shortest path distance in G from node i to node j. Suppose there are AFV drivers willing to travel between OD pairs in G. An AFV demand q is defined as a five-tuple $\langle o_q, d_q, f_q, r_q, \lambda_q \rangle$, where

o_q and d_q are the origin and destination nodes, respectively; f_q is the flow volume; r_q is the range of the vehicle; and λ_q is the total distance that drivers can tolerate. With this definition, we can incorporate the vehicle flows between the same OD pair with different deviation tolerances. The set of demands is denoted by Q. Similar to previous studies, an AFV is assumed to depart from its origin with a half-full tank and is required to arrive at its destination at least with a half-full tank unless these nodes have AFSs. If there exists a station at the origin, then the AFV departs from the origin with a full tank. If the destination has a station, then the AFV can arrive at this node with an empty tank. The logic behind such an assumption is related to round trips between OD pairs; please refer to Kuby and Lim (2005) and MirHassani and Ebrazi (2013) for further details.

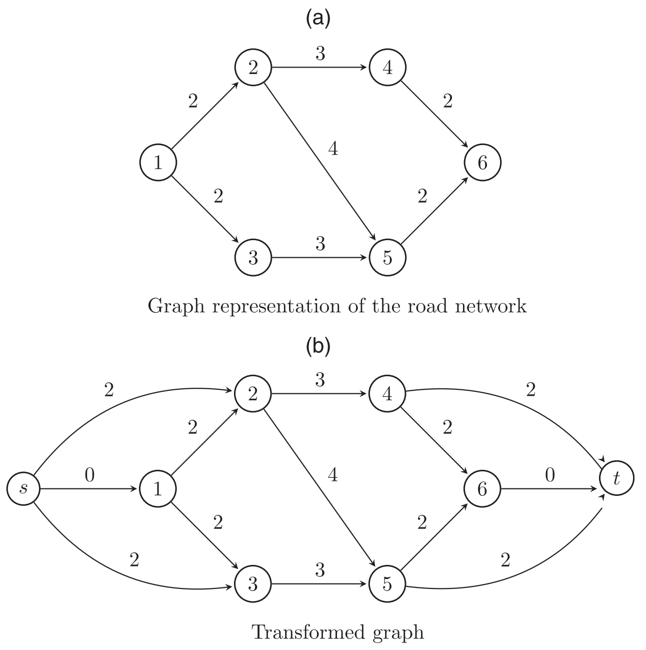
Definition 1. The RSLP-R is defined as finding a subset of N with cardinality at most p to locate the refueling stations such that the total amount of refueled vehicle flow respecting range and tolerance limitations is maximized.

A node is called a *dominated node* for a demand q if it does not appear on any path from o_q to d_q of length at most λ_q. Note that we can identify whether a node i is dominated by checking the shortest path distance between o_q and d_q using node i. Having $\delta_{o_q i} + \delta_{i d_q} > \lambda_q$ implies that node i is a dominated node for demand q. Next, we adapt the network transformation of MirHassani and Ebrazi (2013) to the case in which OD paths are not fixed. For each $q \in Q$, consider graph $G_q = (N_q, A_q)$ with node set $N_q = \{s_q, t_q\} \cup \{i \in N : \delta_{o_q i} + \delta_{i d_q} \le \lambda_q\}$, where s_q and t_q are two new dummy nodes, and arc set $A_q = A_q^1 \cup A_q^2 \cup A_q^3$, where

$$A_q^1 = \{(s_q, j) : \delta_{o_q j} \le r_q/2, j \in N_q \backslash \{s_q, t_q\}\},$$
$$A_q^2 = \{(i, t_q) : \delta_{i d_q} \le r_q/2, i \in N_q \backslash \{s_q, t_q\}\},$$
$$A_q^3 = \{(i, j) : \delta_{ij} \le r_q, i, j \in N_q \backslash \{s_q, t_q\}, i \ne j\}.$$

Arc $(s_q, j) \in A_q^1$ has length $\delta_{o_q j}$, arc $(i, t_q) \in A_q^2$ has length $\delta_{i d_q}$, and arc $(i, j) \in A_q^3$ has length δ_{ij}. Each arc in the transformed graph corresponds to traveling on the shortest path in the underlying road network between the tail and the head nodes of the arc without refueling. The dummy nodes s_q and t_q are added to model the refueling logic, and they represent departing from the origin with a half-full tank and arriving at the destination with at least a half-full tank. Therefore, the arcs emanating from s_q (i.e., arcs in set A_q^1) and the arcs entering to t_q (i.e., arcs in set A_q^3) have length at most $r_q/2$. Refueling at the origin node is represented by arc (s_q, o_q). Traversing this arc is identified with the AFV departing from node o_q with a full tank. The same logic also applies to the destination node. All nodes except the first and the last nodes on a given path are referred to as the path's *internal nodes*. By assumption, an AFV departs from s_q with a half-full tank, and no station is required to traverse an arc emanating from this node. By construction, all other arcs in the transformed graph can be traversed if an AFS is located at the tail node of the arc. Therefore, to satisfy a given demand q, there should exist a path in G_q of distance at most λ_q from s_q to t_q with a refueling station at each of its internal nodes. We refer to such a path as *feasible*.

Figure 1. Length-Bounded Cut Example



For a demand $q \in Q$, let $\mathcal{P}_q$ be the set of all directed paths in G_q from s_q to t_q with lengths at most λ_q. A subset of nodes $S \subseteq N_q \setminus \{s_q, t_q\}$ is called a *q-node-cut* for path set $\mathcal{P}_q$ if S intersects each path in $\mathcal{P}_q$ at a node different from s_q and t_q; in other words, removing S from the set of nodes disconnects s_q and t_q. A q-node-cut S is called *minimal* for $\mathcal{P}_q$ if no proper subset of S is a q-node-cut for $\mathcal{P}_q$. Let Γ_q represent the set of all q-node-cuts for $\mathcal{P}_q$.

Figure 1, (a) and (b), displays a graph representation of an example road network and its transformed graph, respectively. Consider a demand q with $o_q = 1$, $d_q = 6$, $r_q = 4$, and $\lambda_q = 7$. The cuts $\{2,3\}, \{2,5\}, \{3,4\}$, and $\{4,5\}$ in the transformed graph are all minimal q-node-cuts for $\mathcal{P}$. Note that although cuts $\{2,3\}, \{2,5\}$, and $\{4,5\}$ disconnect all paths from s_q to t_q, cut $\{3,4\}$ disconnects only the length-bounded paths, and the s_q-(1)-2-5-(6)-t_q path remains connected.

2.1. Natural Formulation

In this section, we propose a formulation based on the notion of q-node-cuts. The model has the following variables:

$$x_i = \begin{cases} 1, & \text{if there is arefueling station located at node } i \in N \\ 0, & \text{otherwise;} \end{cases}$$

$$y_q = \begin{cases} 1, & \text{if afeasible path is constructed for demand } q \in Q \\ 0, & \text{otherwise.} \end{cases}$$

We refer to x as *location variables* and y as *cover variables*. For convenience, we use $x(S) = \sum_{i \in S} x_i$ for set $S \subseteq N$. We formulate the RSLP-R as follows:

$$\max \sum_{q \in Q} f_q y_q \tag{1}$$

$$\text{s.t. } x(N) \le p, \tag{2}$$

$$y_q \le x(S) \qquad \forall q \in Q, S \in \Gamma_q, \tag{3}$$

$$x_i \in \{0,1\} \qquad \forall i \in N, \tag{4}$$

$$y_q \in \{0,1\} \qquad \forall q \in Q. \tag{5}$$

The objective function maximizes the total vehicle flow refueled. Constraint (2) ensures that at most p stations are located. Constraints (3) express the fact that for having a feasible trip for demand q, the corresponding transformed graph needs to be connected from s_q to t_q by at least one path of length at most λ_q on which every internal node has a refueling station. Suppose that, for demand q, there exists no such path. Then there exists a subset S of $N_q \setminus \{s_q, t_q\}$ such that S contains at least one node from each path in $\mathcal{P}_q$. If none of the nodes in S has a station, constraints (3) for this choice of S force y_q to zero because q cannot be refueled. Constraints (4) and (5) are integrality constraints. Note that one can relax integrality of the cover variables without changing the optimal value. One of the main advantages of this model is that we have natural variables associated with only the nodes and OD pairs.

The idea of q-node-cuts can easily be extended to the set-covering version of the problem, in which one seeks to find the minimum number of AFSs to satisfy *all* the demand.

$$\min \sum_{i \in N} x_i \tag{6}$$

$$\text{s.t. } x(S) \ge 1 \qquad \forall q \in Q, S \in \Gamma_q, \tag{7}$$

$$x_i \in \{0,1\} \qquad \forall i \in N. \tag{8}$$

The objective function minimizes the number of selected refueling stations. Constraints (7) ensure that, for every demand, all q-node-cuts have at least one selected refueling station, which implies that there exists a length-bounded path between every OD pair. Constraints (8) are variable restrictions.

The budget for AFSs is usually limited. For this reason, we consider the maximum-covering version of the problem in this paper.

2.2. Polyhedral Analysis

Let $\mathcal{X}$ be the feasible set of the natural formulation given by (2)–(5). In the following, we assume that $|N| \ge 2$. For a path $\pi \in \mathcal{P}_q$, let $N(\pi)$ be the set of its internal nodes. We define, for each $q \in Q$, set $\mathcal{P}'_q$ to be the set of paths in $\mathcal{P}_q$ with at most p internal nodes. We assume, without loss of generality, that $\mathcal{P}'_q$ contains at least one path for all $q \in Q$ (otherwise, $y_q = 0$ in all feasible solutions, and demand q can be removed from set Q).

Let $conv(\mathcal{X})$ be the convex hull of all the solutions in $\mathcal{X}$. In this section, we study the polyhedral properties of $conv(\mathcal{X})$ and prove that most of the constraints of the natural formulation are facet-defining inequalities under some conditions. The proofs of Propositions 1 through 6 are presented in the online appendix.

Proposition 1. The convex hull of $\mathcal{X}$ *is full dimensional.*

Next, we give necessary and sufficient conditions for the trivial inequalities to be facet defining.

Proposition 2. *For $q \in Q$, the inequality $y_q \ge 0$ is facet defining for $conv(\mathcal{X})$.*

Proposition 3. *For $i \in N$, the inequality $x_i \ge 0$ is facet defining for $conv(\mathcal{X})$ if and only if $\{i\}$ is not a q-node-cut for $\mathcal{P}'_q$ for all $q \in Q$.*

Proposition 4. *For $i \in N$, the inequality $x_i \le 1$ is facet defining for $conv(\mathcal{X})$ if and only if $p \ge 2$ and there exists a path $\pi^q \in \mathcal{P}'_q$ such that $|N(\pi^q) \cup \{i\}| \le p$ for all $q \in Q$.*

In the next two propositions, we put forward the conditions under which the other constraints of the model are facet defining.

Proposition 5. *The inequality $x(N) \le p$ is facet defining for $conv(\mathcal{X})$ if and only if $p < |N|$.*

Proposition 6. *For $q \in Q$ and a q-node-cut S that is minimal for path set $\mathcal{P}'_q$, the inequality $y_q \le x(S)$ is facet defining for $conv(\mathcal{X})$ if and only if, for every $\hat{q} \in Q \setminus \{q\}$, either there exists a path $\pi^{\hat{q}} \in \mathcal{P}'_{\hat{q}}$ with $N(\pi^{\hat{q}}) \cap S = \emptyset$ or there exist a node $i \in S$, a path $\pi^q \in \mathcal{P}'_q$, and a path $\pi^{\hat{q}} \in \mathcal{P}'_{\hat{q}}$ such that $N(\pi^q) \cap S = N(\pi^{\hat{q}}) \cap S = \{i\}$ and $|N(\pi^q) \cup N(\pi^{\hat{q}})| \le p$.*

3. Separation Problem

In our formulation, constraints (3) are exponential in number, and we need a cutting plane algorithm to generate them. For a given solution (x^*, y^*) and a demand $q \in Q$ with $y^*_q > 0$, the separation problem is to identify a q-node-cut $S \subseteq N_q \setminus \{s_q, t_q\}$ for path set $\mathcal{P}_q$ with $x^*(S) < y^*_q$ or to conclude that none exists.

3.1. Separating Integer Solutions

Consider an integer solution (x^*, y^*). For a subset of nodes $N' \subseteq N$, we define $G_q(N')$ to be the subgraph of G_q induced by nodes in N'. For each $q \in Q$ with $y^*_q = 1$, we compute the length of the shortest path from s_q to t_q

in $G_q(N_q^*)$, where $N_q^* = \{s_q, t_q\} \cup \{i \in N_q : x_i^* = 1\}$. If the shortest path length is less than or equal to λ_q, then the solution contains a feasible path for demand q. If not, then at least one station needs to be located at those nodes with $x_i^* = 0$ to refuel demand q. In other words, inequality (3) for the q-node-cut $N_q \setminus N_q^*$ is violated. We refer to the process of identifying such a q-node-cut as *IntSep*.

Clearly, $N_q \setminus N_q^*$ may not be a minimal q-node-cut for path set $\mathcal{P}_q$. For a node $i \in N_q \setminus N_q^*$, if the shortest path from s_q to t_q in $G_q(N_q^* \cup \{i\})$ is greater than λ_q, then locating a station at node i cannot render the demand feasible, and $N_q \setminus (N_q^* \cup \{i\})$ is also a q-node-cut for $\mathcal{P}_q$. The associated inequality (3) for this new q-node-cut dominates the former one. We can repeat this operation until a minimal q-node-cut is attained. When the cuts generated by IntSep are also minimalized, then the process is referred to as *IntSep-M*. In the following section, we show that strengthening the generated cuts in this fashion proved to be highly effective in reducing the computational times.

3.2. Separating Fractional Solutions

Now suppose that (x^*, y^*) is fractional. Consider a demand $q \in Q$ with $y_q^* > 0$. We define the minimum weight q-node-cut problem (*MqCP*) as follows: given graph G_q with node weight x_i^* for node $i \in N_q \setminus \{s_q, t_q\}$, find a minimum weight subset S^* of $N_q \setminus \{s_q, t_q\}$ such that deleting the nodes in S^* disrupts all directed paths from node s_q to node t_q with lengths of at most λ_q. There exists an inequality (3) for demand q that is violated by (x^*, y^*) if and only if $x^*(S^*) < y_q^*$.

The special case of MqCP in which all arcs have unit lengths is called the *length-bounded minimum node-cut problem* (Lovász et al. 1978), which is known to be NP-hard for lengths greater than four units (Baier et al. 2006, Mahjoub and McCormick 2010).

Consider a variable u_i, which equals one if node $i \in N_q \setminus \{s_q, t_q\}$ is in the q-node-cut and zero otherwise. We also define π_i to be the length of a shortest path from node $i \in N_q$ to node t_q in graph $G_q(N_q^*)$, where $N_q^* = \{s_q, t_q\} \cup \{i \in N_q \setminus \{s_q, t_q\} : u_i = 0\}$. We let M be a very large number and ε a very small positive number. We refer to the following model as the *minimum weight q-node-cut model* (MqCM):

$$(\text{MqCM}) \quad \min \sum_{i \in N_q \setminus \{s_q, t_q\}} x_i^* u_i \tag{9}$$

$$\text{s.t. } \pi_{t_q} = 0, u_{t_q} = 0, \tag{10}$$

$$\pi_i \le \pi_j + \delta_{ij} + M u_j \quad \forall (i,j) \in A_q, \tag{11}$$

$$\pi_{s_q} \ge \lambda_q + \varepsilon, \tag{12}$$

$$\pi_i \ge 0 \quad \forall i \in N_q, \tag{13}$$

$$u_i \in \{0,1\} \quad \forall i \in N_q \setminus \{s_q, t_q\}. \tag{14}$$

Table 1. B&C Algorithm Implementations

	Separation algorithm	
Implementation	Integer solutions	Fractional solutions
B&C-1	IntSep-M	
B&C-2	IntSep-M	MqCM
B&C-3	IntSep-M	MinCut
B&C-4	IntSep	
B&C-5	IntSep-M	MinCut-M

The objective function minimizes the total weight of nodes in the node cut. Constraint (10) sets the shortest path length from t_q to itself to zero, and it forbids this node to be in the node cut. Constraints (11) ensure that π_i is not more than the length of a shortest path from node i to node t_q in the graph obtained by removing the nodes with $u_j = 1$. The shortest path length from s_q to t_q after these nodes are removed is forced to be greater than λ_q by constraint (12). Constraints (13) and (14) are the nonnegativity and integrality constraints.

In the next section, we present computational results in which we use this model for separation. However, this turns out to be computationally costly in most cases. For this reason, we also propose a simple and efficient heuristic approach. Observe that any node cut that disconnects s_q and t_q is also a q-node-cut for set $\mathcal{P}_q$. For all $q \in Q$, we search for the minimum weight node cut in graph G_q and add the corresponding cut if a violation is identified. To find a node cut S, we split every node $i \in N_q \setminus \{s_q, t_q\}$ into two nodes i' and i'' and add arc (i', i'') for every such i. Those arcs entering into node i will be entering into node i', and those arcs leaving node i will be leaving node i''. All the arcs have infinite capacity except for those that represent nodes (i.e., arcs of the form (i', i'')). Arc (i', i'') has capacity equal to x_i^*. Solving a minimum cut problem on this transformed network gives a cut with the minimum $x^*(S)$ value. If $x^*(S) < y_q^*$, then a violated cut is identified that separates a fractional or integer infeasible solution at hand. We refer to this process as the *mincut* heuristic. Note that the minimum node cuts we obtain by the heuristic are not necessarily minimal q-node-cuts because they disconnect all paths regardless of their lengths. Therefore, we can strengthen the generated cuts by the cut minimalization process, similar to the logic in integer separation. We refer to the heuristic as *mincut-M* if the cuts generated by the mincut are minimalized.

4. Computational Study

We propose a B&C algorithm to solve the RSLP-R because constraints (3) are exponential in number. We tested five different implementations (Table 1). The first one, which we refer to as *B&C-1*, uses separation only at integer solutions. In the second and third implementations, we separate both integer and fractional solutions. We

Table 2. Characteristics of Instances

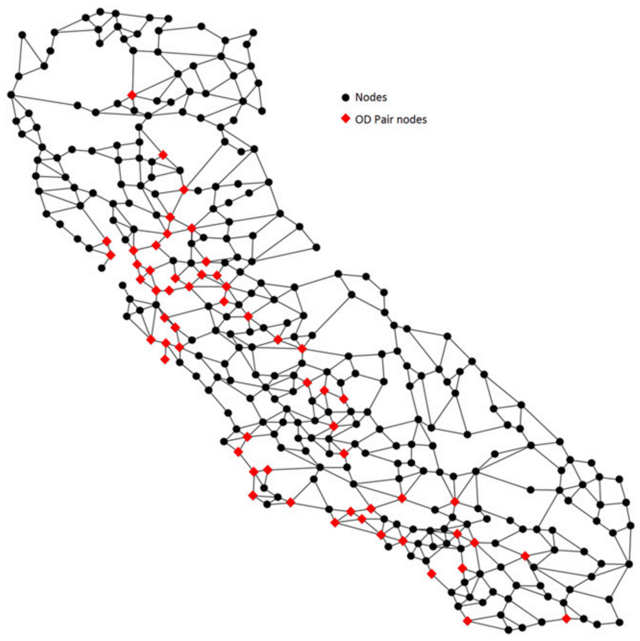
			Node degree			OD pairs		
Network	Number of nodes	Number of arcs	Minimum	Mean	Maximum	Minimum distance	Mean distance	Maximum distance
CA	339	1,234	2	3.64	14	30.06	153.37	463.50
G-250	250	636	2	5.09	14	30.02	138.93	389.00
G-500	500	1,284	2	5.14	20	30.05	142.95	366.72
G-750	750	1,922	2	5.13	16	30.00	153.87	522.43
G-1000	1,000	2,580	2	5.16	22	30.00	160.98	458.86

use the MqCM in B&C-2 and the mincut heuristic in B&C-3 to separate fractional solutions. B&C-4 and B&C-5 are designed to test the efficiency of the cut minimalization process. In B&C-4, we only separate integer solutions, similar to B&C-1, but without minimalizing the generated cuts. B&C-5, by contrast, is designed to test whether the cut minimalization can also help in strengthening the cuts we obtain by the mincut heuristic. In B&C-5, we separate both integer and fractional solutions, similar to B&C-3, with the only difference that the cuts generated by the mincut heuristic are minimalized.

For branching, we use the default settings of CPLEX. We turn off the CPLEX cuts as our preliminary analysis showed that this gives shorter computation times.

Extensive computational experiments are carried out to test the efficiency of the proposed B&C algorithms. The experiments are executed using CPLEX 12.6.1 (IBM 2014), implemented in a Java programming environment under Linux using Concert Technology. The computer has an Intel Xeon E5-2630 v2 processor at 2.60 GHz and 96 GB of RAM. The algorithms are implemented using callback classes. A time limit of one hour is set in all implementations.

Figure 2. (Color online) California Road Network



Characteristics of network topologies considered in this study are detailed in Table 2. CA is a real-world representation of the California road network with 339 nodes and 1,234 arcs, as shown in Figure 2 (Arslan et al. 2014). The nodes of the network represent road intersections or urban population centers. Similar to previous studies, all urban centers with a population of 50,000 or more are selected as origin or destination nodes, which are depicted in Figure 2 as OD pair nodes. There are 1,167 OD pairs, and they are 30 kilometers or more apart from each other. The vehicle flow volume between each OD pair is calculated according to the gravity model by Hodgson (1990). Networks G-250, G-500, G-750, and G-1000 are randomly generated networks with 250, 500, 750, and 1,000 nodes, respectively. To generate random graphs, we use JGraphT Java graph library (Naveh et al. 2008). Arc lengths are generated from a uniform distribution on the interval (0, 50) kilometers. Triangular inequality is not considered. For these graphs, we randomly select nodes with equal probabilities to represent origins or destinations. Similar to the CA network, we consider those OD pairs that are 30 kilometers or more apart. The number of OD pairs changes between experiments and varies between 1,000 and 4,000.

We mainly compare our results with the results of the B&P algorithm by Yıldız et al. (2016). Kim and Kuby (2013) present a heuristic algorithm based on network transformation; however, because the refueling station location problem is strategic in nature, we prefer to compare only with the exact solution methods in the literature. To this end, a computational comparison of the model by Kim and Kuby (2012) with a B&P algorithm is previously presented by Yıldız et al. (2016), and it is shown that the former model cannot scale up to large networks because of enumeration requirements.

Let $\hat{\lambda}_q = 100 \times \lambda_q / \delta_{o_q,d_q}$ for all $q \in Q$ be the deviation tolerance as a percentage of the shortest path length. Note that our model is capable of handling different driver tolerances for each OD pair. Furthermore, our demand definition allows us to model varying tolerances between the same OD pair. However, as in the previous studies, we assume equal deviation tolerance percentage for all demand, which we refer to as $\hat{\lambda} = \hat{\lambda}_q$, $q \in Q$.

Table 3. Performance Comparison of the B&C Algorithms

		Sol time, s					Root node gap, %					nNodes					nCuts				
p	Tol, %	B&C-1	B&C-2	B&C-3	B&C-4	B&C-5	B&C-1	B&C-2	B&C-3	B&C-4	B&C-5	B&C-1	B&C-2	B&C-3	B&C-4	B&C-5	B&C-1	B&C-2	B&C-3	B&C-4	B&C-5
1	0	1.9	2.0	1.9	1.6	1.9	0	0	0	0	0	0	0	0	0	0	1,183	1,183	1,183	1,196	1,183
	10	7.7	7.3	7.9	7.2	7.3	0	0	0	0	0	0	0	0	0	0	1,173	1,173	1,173	1,207	1,173
	20	14.9	11.1	11.5	10.9	11.4	0	0	0	0	0	0	0	0	0	0	1,173	1,173	1,173	1,174	1,173
	50	20.3	20.6	20.4	19.9	20.8	0	0	0	9.83	0	0	0	0	16	0	1,200	1,200	1,200	1,560	1,200
5	0	3.4	3.3	3.5	3,601.9[a]	3.5	0	0	0	N/A	0	0	0	0	4,765	0	2,408	2,408	2,408	179,927	2,408
	10	29.9	230.2	30.8	3,607.0[a]	31.7	0.38	0	0	N/A	0	5	0	0	2,637	0	3,247	4,084	3,189	215,873	3,189
	20	92.0	88.4	105.8	3,615.8[a]	250.2	4.82	2.55	2.55	N/A	2.55	34	34	23	1,931	17	3,818	3,818	5,237	243,382	4,034
	50	119.3	123.4	135.9	3,623.1[a]	330.8	0.50	0.32	0.32	N/A	0.32	9	9	7	1,535	6	2,712	2,712	2,720	391,461	2,753
10	0	4.1	54.9	4.0	3,601.6[a]	4.8	0.78	0.60	0.60	N/A	0.60	5	3	3	6,423	3	2,362	2,870	2,427	122,227	2,226
	10	25.3	24.3	28.4	3,608.1[a]	50.2	1.18	1.18	1.18	N/A	1.18	22	22	6	4,200	6	2,478	2,478	2,418	160,256	2,416
	20	49.5	3,623.6[a]	107.4	3,612.4[a]	494.6	0.39	N/A	0.25	N/A	0.24	25	0	40	3,704	30	2,630	6,648	2,779	230,458	2,467
	50	90.7	90.9	108.8	3,621.3[a]	281.4	0.25	0.14	0.14	N/A	0.14	25	25	9	3,921	8	1,952	1,952	2,050	233,232	2,054
15	0	4.1	1,020.7	3.7	3,601.8[a]	4.8	0.06	0	0	N/A	0	7	0	0	5,310	0	2,877	3,863	2,865	108,078	2,829
	10	21.2	21.5	21.4	3,607.7	20.0	0.01	0.01	0.01	N/A	0.01	4	4	4	8,563	0	1,938	1,938	1,938	102,612	1,938
	20	44.6	3,646.7[a]	63.3	3,612.1[a]	208.0	0.08	N/A	0.07	N/A	0.07	7	0	8	17,552	8	2,019	4,005	2,483	66,371	1,648
	50	72.2	641.0	81.5	236.9	145.8	0	0	0	0	0	0	0	0	1,536	0	1,392	1,403	1,372	50,886	1,372
20	0	4.5	676.4	10.7	3,601.7[a]	43.4	0.48	0.34	0.34	N/A	0.34	72	66	30	13,389	42	2,556	6,391	3,401	66,353	3,332
	10	17.8	581.9	24.6	286.8	65.1	0.02	0.02	0.02	0.16	0.02	5	11	3	4,466	3	1,680	2,136	1,760	18,334	1,461
	20	35.1	3,660.3[a]	97.1	1,002.0	306.2	0.02	N/A	0.02	0.03	0.02	45	0	32	22,452	11	1,446	1,869	1,437	15,651	1,535
	50	70.8	72.2	71.3	71.5	71.7	0	0	0	0	0	0	0	0	217	0	1,295	1,295	1,295	17,770	1,295
25	0	6.1	338.4	60.2	3,601.7[a]	208.7	0.12	0.12	0.12	N/A	0.12	247	241	383	89,833	223	3,217	2,967	2,869	23,058	2,952
	10	16.4	16.4	26.2	14.3	36.4	0.01	0.01	0.01	0.01	0.01	10	10	11	173	3	1,284	1,284	1,331	5,782	1,330
	20	31.4	446.1	47.4	25.9	93.0	0	0	0	0	0	5	5	5	31	0	1,234	1,234	1,277	4,096	1,234
	50	72.3	71.6	71.5	38.3	74.1	0	0	0	0	0	0	0	0	10	0	1,297	1,297	1,297	5,203	1,297
30	0	4.7	88.4	3.9	12.1	5.7	0	0	0	0	0	16	0	0	629	0	2,456	1,889	1,970	7,831	1,907
	10	15.1	14.7	14.8	13.0	14.3	0	0	0	0	0	0	0	0	45	0	1,215	1,215	1,215	3,809	1,215
	20	38.6	33.8	34.0	15.9	35.1	0	0	0	0	0	0	0	0	3	0	1,283	1,283	1,283	1,900	1,283
	50	72.0	71.0	73.3	39.6	73.2	0	0	0	0	0	0	0	0	6	0	1,297	1,297	1,297	3,353	1,297
35	0	2.9	2.9	3.1	3.4	2.9	0	0	0	0	0	2	2	2	11	2	1,344	1,344	1,344	2,588	1,344
	10	13.9	14.9	14.9	9.2	14.7	0	0	0	0	0	0	0	0	0	0	1,184	1,184	1,184	1,562	1,184
	20	34.4	33.2	33.2	21.3	33.3	0	0	0	0	0	0	0	0	0	0	1,283	1,283	1,283	2,492	1,283
	50	74.2	72.7	70.1	54.7	72.2	0	0	0	0	0	0	0	0	0	0	1,297	1,297	1,297	3,700	1,297
Averages		34.7	493.9	43.5	1,525.0	94.3	0.28	N/A	0.16	N/A	0.16	17.0	13.5	17.7	6,042.4	11.3	1,873	2,255	1,942	71,668	1,853

[a]The algorithm terminated because of the time limit.

Table 4. Performance Summaries of B&C Algorithms

	Average time, s				Nodes removed, %	
Implementation	Total	Integer	Fractional	AvgNCuts	IntSep-M	MinCut-M
B&C-1	34.7	34.3	0	1,873	65.11	—
B&C-2	493.9	19.0	473.9	2,255	64.82	—
B&C-3	43.5	30.9	12.2	1,942	64.71	—
B&C-4	1,525.0	43.6	0	71,668	—	—
B&C-5	94.3	28.7	65.2	1,853	64.75	17.66

In the following, we first compare the five B&C algorithms on the CA network instances. This network is the largest-size network used in testing the B&P algorithm in Yıldız et al. (2016). We then increase the number of demands and the size of the networks to investigate the limitations of our approach.

4.1. California Network

Our first objective is to compare five B&C implementations in terms of solution times using this large network. For this purpose, we consider a vehicle with a range of 100 kilometers. As in Yıldız et al. (2016), the number of stations considered are $1, 5, 10, \ldots, 35$, and the drivers are assumed to be 0%, 10%, and 20% tolerant to deviating from their shortest paths. Different from the previous settings, we also consider deviations of 50% in our experimental design. The results are presented in Table 3. The two leftmost columns are parameters of the experiment: p is the number of stations to be located, and "Tolerance, %" is the drivers' deviation tolerance from the shortest path as a percentage. For instance, 10% tolerance means that the drivers are willing to drive up to 10% more from their shortest paths. The following five columns in the table show the solution times in seconds. "Root node gap" is the percentage gap between the upper bound (UB) at the root node and the optimal value, calculated as $(\mathrm{UB} - \mathrm{OPT})/\mathrm{OPT} \times 100\%$. Note that for the B&C-2 algorithm, the upper bound we obtain at the root node is equal to the optimal value of the linear programming (LP) relaxation of the model because we use exact algorithms for separation of both integer and fractional solutions and turn off the presolve and CPLEX cuts. "nNodes" is the number of nodes in the search tree. The rightmost five columns show the number of user cuts added by the algorithms.

The average solution times are 34.7, 493.9, 43.5, 1,525.0, and 94.3 seconds for the B&C-1 through B&C-5 algorithms, respectively. In Table 4, the time for separating integer and fractional solutions are shown in columns (3) and (4), respectively. The average number of cuts added (*AvgNCuts*) is reported in column (5). The average percentage of nodes removed from cuts by the minimalization algorithm for integer separation (*IntSep-M*) and the mincut heuristic (*MinCut-M*) are also reported in the table.

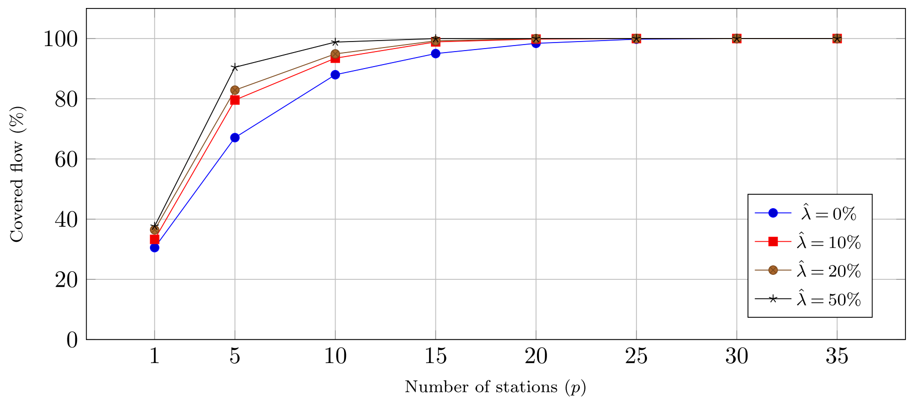
The B&C-2 algorithm could not find the optimal solution of three instances (marked in Table 3) within the one-hour time limit; it terminated at the root node, and the maximum optimality gap was 3.1%. Table 4 shows that the B&C-2 algorithm spends more than 95% of the time for the fractional separation at the root node; however, no significant improvements can be achieved over the other algorithms in terms of root node gap. In other words, the time that the B&C-2 algorithm spends to separate inequalities (3) exactly does not pay off. Furthermore, the unpredictable solution times of the *MqCM* model to separate fractional solutions in B&C-2 cause extended solution times in several instances, and therefore, the solution times do not follow an obvious pattern in Table 3. Next, we compare the B&C-1 and B&C-4 algorithms and observe that the cut-minimalizing algorithm is highly effective in reducing the computation times. Without minimalization, the B&C-4 algorithm spends less than 3% of the time for solving the separation problem, mainly because of weak cuts being added. Therefore, it fails to solve 13 of the 32 instances within the time limit (Table 3). In B&C-1, by contrast, the minimalization process removes, on average, 65.11% of the nodes from the cuts generated by the IntSep-M algorithm (Table 4). Finally, we compare the B&C-3 and B&C-5 algorithms to see the effects of cut minimalization in the cuts generated by the mincut heuristic. In B&C-5, on average, minimalization removes 17.66% of the nodes from the cuts generated by the heuristic; however, this came at a cost of multiplying the fractional separation times by more than five. The average time for fractional separation increased from 12.2 to 65.2 seconds. Even though the average number of cuts added is reduced from 1,942 to 1,853, the average solution time increased from 43.5 seconds in B&C-3 to 94.3 seconds in B&C-5. The main reason for the cut minimalization to perform well in the IntSep algorithm but not in the mincut heuristic is that the cuts generated by the mincut heuristic are already small in size. However, the size of q-node-cut generated by the IntSep algorithm is large because all those nodes without a refueling station are in the cut. Thus, more nodes are removed from the cuts generated by the IntSep algorithm, and the time spent for cut minimalization pays off. According to the performance results,

Table 5. Results for the CA Instances with Different Vehicle Ranges

		Range = 100 km					Range = 150 km					Range = 200 km				
p	Tol, %	Opt value, %	Sol time, s	Root node gap, %	nNodes	nCuts	Opt value, %	Sol time, s	Root node gap, %	nNodes	nCuts	Opt value, %	Sol time, s	Root node gap, %	nNodes	nCuts
1	0	30.54	1.9	0	0	1,183	33.95	1.5	0	0	1,208	36.46	2.3	0	0	1,210
	10	33.29	7.7	0	0	1,173	34.62	8.2	0	0	1,227	37.22	8.4	0	0	1,241
	20	36.46	14.9	0	0	1,173	36.83	14.0	0	0	1,284	38.08	22.1	0	0	1,309
	50	37.66	20.3	0	0	1,200	42.19	38.6	0	0	1,333	43.84	44.7	0	0	1,569
5	0	67.08	3.4	0	0	2,408	79.94	3.0	0	0	1,849	85.18	2.7	0	0	1,500
	10	79.57	29.9	0.38	5	3,247	85.91	23.9	0	0	1,911	90.53	17.8	0.26	5	1,545
	20	82.86	92.0	4.82	34	3,818	89.08	52.9	0.17	3	1,927	93.87	48.5	0	0	1,353
	50	90.46	119.3	0.5	9	2,712	94.51	105.5	0	0	1,713	97.52	100.2	0	0	1,385
10	0	87.98	4.1	0.78	5	2,362	92.98	3.3	0.91	28	1,792	95.64	3.5	0.35	7	1,604
	10	93.47	25.3	1.18	22	2,478	97.4	18.6	0	0	1,557	98.63	17.1	0	0	1,386
	20	94.9	49.5	0.39	25	2,630	98.29	46.1	0.07	7	1,655	99.02	48.6	0.04	6	1,355
	50	98.82	90.7	0.25	25	1,952	99.8	91.7	0	0	1,394	99.9	90.1	0	0	1,281
15	0	95.01	4.1	0.06	7	2,877	98.35	4.0	0.28	37	1,872	99.22	3.5	0.25	66	1,482
	10	98.89	21.2	0.01	4	1,938	99.79	19.3	0.05	10	1,315	99.87	16.0	0.06	11	1,182
	20	99.24	44.6	0.08	7	2,019	99.95	45.9	0	0	1,352	99.97	47.4	0.01	2	1,220
	50	100	72.2	0	0	1,392	100	83.9	0	0	1,188	100	101.2	0	0	1,178
20	0	98.41	4.5	0.48	72	2,556	99.89	4.1	0.06	92	1,912	99.97	3.9	0.03	36	1,424
	10	99.82	17.8	0.02	5	1,680	99.98	17.5	0.01	36	1,234	100	17.4	0	0	1,198
	20	99.97	35.1	0.02	45	1,446	100	41.3	0	0	1,176	100	45.3	0	0	1,177
	50	100	70.8	0	0	1,295	100	82.8	0	0	1,188	100	91.2	0	0	1,178
25	0	99.79	6.1	0.12	247	3,217	100	3.0	0	0	1,356	100	2.9	0	0	1,244
	10	99.99	16.4	0.01	10	1,284	100	16.0	0	0	1,201	100	15.8	0	0	1,174
	20	100	31.4	0	5	1,234	100	43.1	0	0	1,171	100	42.1	0	0	1,177
	50	100	72.3	0	0	1,297	100	82.9	0	0	1,188	100	90.2	0	0	1,178
30	0	100	4.7	0	16	2,456	100	3.1	0	0	1,670	100	2.5	0	0	1,205
	10	100	15.1	0	0	1,215	100	17.5	0	0	1,198	100	17.4	0	0	1,169
	20	100	38.6	0	0	1,283	100	40.6	0	0	1,171	100	42.7	0	0	1,177
	50	100	72.0	0	0	1,297	100	87.7	0	0	1,188	100	97.4	0	0	1,178
35	0	100	2.9	0	2	1,344	100	2.7	0	0	1,222	100	2.7	0	0	1,205
	10	100	13.9	0	0	1,184	100	16.9	0	0	1,198	100	15.7	0	0	1,169
	20	100	34.4	0	0	1,283	100	44.5	0	0	1,171	100	42.5	0	0	1,177
	50	100	74.2	0	0	1,297	100	87.1	0	0	1,188	100	90.8	0	0	1,178

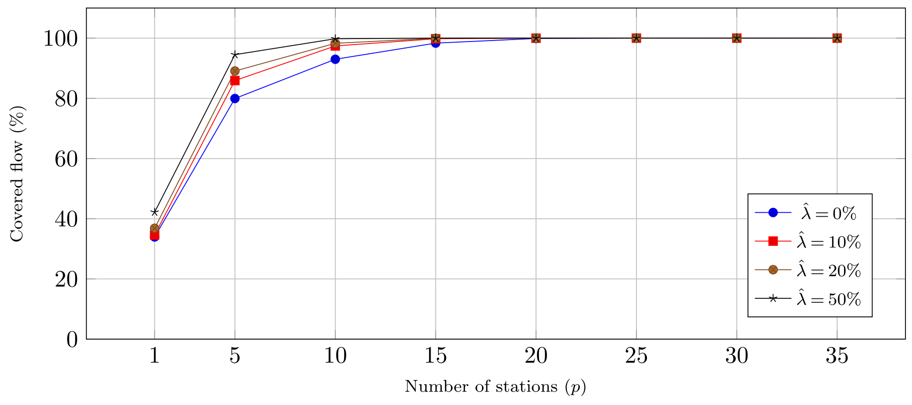
Figure 3. (Color online) Covered Flow as a Percentage of the Total Vehicle Flow for $p = 1, \ldots, 35$ for (a) Range = 100 km, (b) Range = 150 km, and (c) Range = 200 km in the CA network

(a)



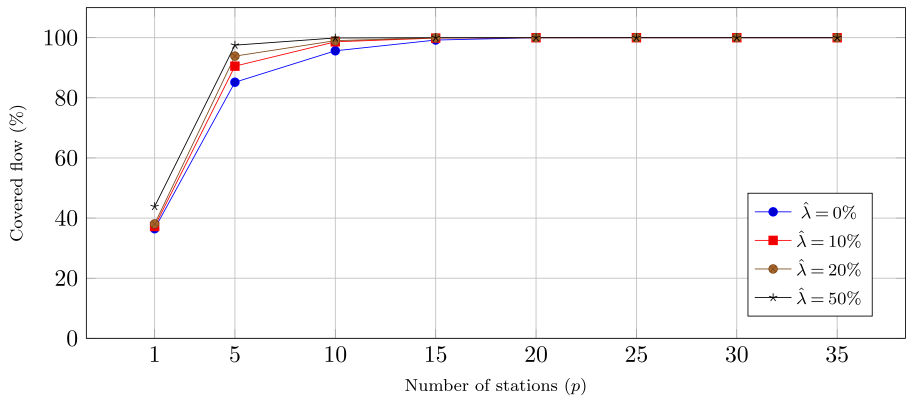
Range = 100 km.

(b)



Range = 150 km.

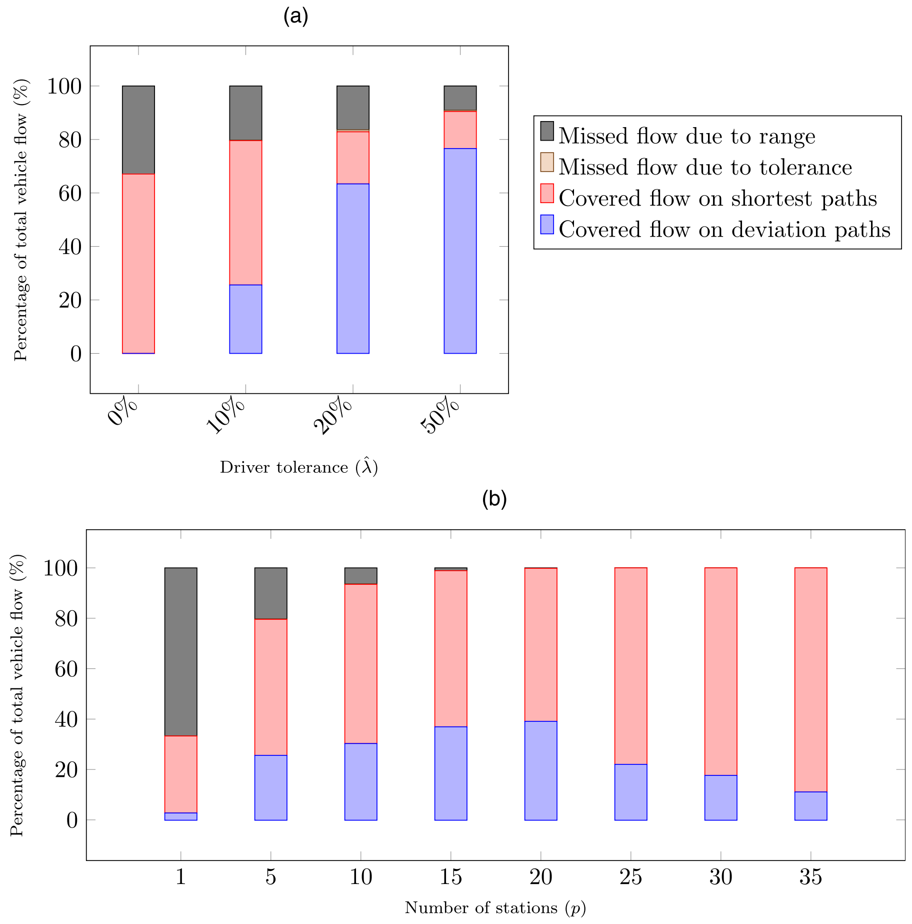
(c)



Range = 200 km.

the B&C-1 algorithm, implementing separation only at integer solutions and the cut minimalization algorithm, performs the best of the five versions. It is more than 14 times faster than the second implementation. The root node gaps of the second and third implementations are the same for all the instances in which both could finish solving the LP relaxation within the time limit. With the best solution times and very small root node gaps, we perform our further analyses using the B&C-1 algorithm.

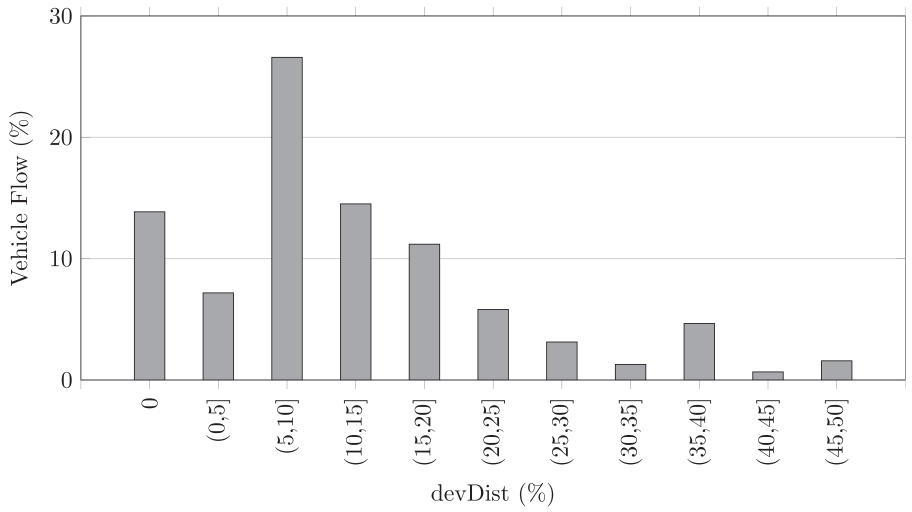
Figure 4. (Color online) (a) Percentage of Covered and Missed Vehicle Flows for Different Driver Tolerances in the California Network with Vehicles of 100-km Range and When $p = 5$ Stations Are Optimally Located, (b) Percentage of Covered and Missed Vehicle Flows When $p = 1, \ldots, 35$ Stations Are Optimally Located in the California Network with Vehicles of 100-km Range and for Deviation Tolerance $\hat{\lambda} = 10\%$



Detailed results with the B&C-1 algorithm and the CA network are presented in Table 5. Ranges of 100, 150, and 200 kilometers are considered. The first two columns present the number of stations to be located and the drivers' tolerance for deviation. Columns (3) through (7) show results for vehicles with a range of 100 kilometers. In addition to the statistics provided in Table 3, column "Opt value, %" shows the percentage of total vehicle flow covered. The following five columns correspond to solutions for vehicles of range 150 kilometers, and the rightmost five columns correspond to those for vehicles of range 200 kilometers. Even though our computer configuration is not the same as the one used in Yıldız et al. (2016), the two are comparable. According to PassMark Software (2017), our computer has a 1.147 times higher CPU mark rating. The results show that our B&C implementation outperforms the previous B&P algorithm: B&C-1 is able to solve the instances even with 50% tolerance in less than two minutes, whereas the B&P implementation could only handle 20% tolerance and terminated before reaching optimality after three hours for several instances.

Figure 3 plots the covered vehicle flow for different deviation tolerances when $p = 1, \ldots, 35$ stations are optimally located and the vehicles have ranges of 100, 150, and 200 kilometers in parts (a), (b), and (c), respectively. Coverage increases to 100% steadily in all three plots. The rate of increase is faster for higher-deviation tolerances. For instance, when range is 100 kilometers in Figure 3(a), in the 0% deviation curve, all demand can be covered when $p = 30$. In the 50% deviation curve, by contrast, full coverage is reached when $p = 15$. The rate of increase is also faster for higher vehicle ranges. When range increases to 200 kilometers in Figure 3(c), in the 0% deviation curve, 100% coverage can now be reached by locating 20 stations rather than 30 stations as in the 100-kilometer scenario in Figure 3(a). Note that the impact of deviation tolerance on the covered flow is more pronounced when the range is shorter. For instance, when $p = 5$ and range is 100 kilometers in Figure 3(a), coverage increases from 67.09% in $\hat{\lambda} = 0\%$ to 90.47% in $\hat{\lambda} = 50\%$ (23.38% difference). The increase is only 12.34% when the range is 200 kilometers in Figure 3(c) from

Figure 5. Distribution of Vehicle Flow Grouped into Deviation Distance Intervals in the CA Network for Vehicles of 100-km Range and When $p = 5$ and $\hat{\lambda} = 50\%$



Note. The deviation distance on the horizontal axis is presented as a percentage of the shortest path.

85.18% in $\hat{\lambda} = 0\%$ to 97.52% in $\hat{\lambda} = 50\%$. By locating only a single station (i.e., $p = 1$) in Los Angeles in the southern part of California, 30.55%–43.85% of the vehicle flow can be covered, depending on the range of the vehicle and deviation tolerance of the drivers. Fifteen stations cover more than 90% of the demand for all range and tolerance levels considered in Figure 3. By locating 30 stations, all vehicle flows can be covered in all settings.

Note that when the problem is solved using our natural formulation, the optimal refueling station locations and the demand that can be covered are given by the x and y variables, respectively. Consider a demand q and the corresponding OD pair (o_q, d_q). We postprocess the optimal solution to determine the path that a vehicle flow takes by solving a shortest-path algorithm in the graph induced by the optimal station locations. Let $\delta^*_{o_q d_q}$ be the shortest path length in the induced graph. We then compare this length with $\delta_{o_q d_q}$, which is the shortest path length in the original road network. If $\delta^*_{o_q d_q} = \delta_{o_q d_q}$, then the vehicle flow travels on its shortest path. If $\delta_{o_q d_q} < \delta^*_{o_q d_q} \leq \lambda_q$, then the vehicle flow deviates from its shortest path to travel to its destination. If $\lambda_q < \delta^*_{o_q d_q} < \infty$, in other words, if the shortest path length is greater than the drivers' tolerance but finite, then the vehicle flow is not covered because there is no path of length at most λ_q. In this case, we refer to such flow as *missed flow because of tolerance*. If $\delta^*_{o_q d_q} = \infty$, that is, there is no path from the origin to the destination and these two nodes are disconnected, then the range is not long enough to travel between the located stations. Therefore, we refer to such flow as *missed flow because of range*.

Figure 4(a) shows the breakdown of covered and missed vehicle flows for different tolerances when five stations are optimally located in the CA network with vehicles of 100-kilometer range. This figure is representative to show the reasons for covering and missing vehicle flows. When the drivers are not tolerant to deviating (i.e., 0% tolerance), then 67.09% of the total vehicle flow can be covered. The coverage increases to 90.46% when the drivers tolerate 50% of their shortest paths. The deviating vehicle flow percentage increases for higher deviation tolerances, and vehicle flow traveling on their shortest paths decreases. We also observe in the figure that the main cause of missing vehicle flow is the limited range. The missed flow because of tolerance is at maximum 0.7% for 20% deviation tolerance in Figure 4(a). It is very rarely the case in all the scenarios that there exists a required infrastructure for vehicles to travel but the drivers are intolerant to deviating.

Figure 4(b) shows the covered and missed flow percentages in the optimal solutions when $p = 1, \ldots, 35$ stations are located in the CA network with vehicles of 100-kilometer range and driver deviation tolerance of 10%. The percentage of the deviating vehicle flow is increasing until full coverage is achieved at 20 stations. Higher numbers of stations being located in the network increases their availability on the shortest paths of the drivers, which leads to less vehicle flow deviating.

We now provide insights about the deviation distances. Similar to the deviation tolerance, we present the deviation distance as a percentage of the shortest path. We refer to the deviation distance as *devDist*, which is given by devDist $= 100 \times (\delta^*_{o_q d_q} - \delta_{o_q d_q})/\delta_{o_q d_q}$ for $q \in Q$. Figure 5 plots the distribution of vehicle flow grouped into devDist intervals in the CA network for vehicles of 100-kilometer range and when $p = 5$ and $\hat{\lambda} = 50\%$. This figure is representative to show the

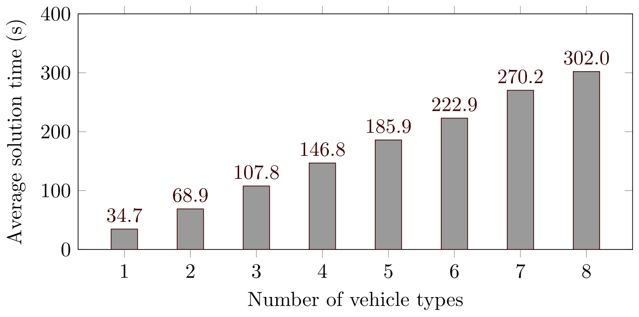
Table 6. Results for the CA Instances with Different Vehicle Types

		0% driver tolerance					10% driver tolerance					20% driver tolerance					50% driver tolerance				
Number of vehicle types	*p*	Opt value, %	Sol time, s	Root node gap, %	nNodes	nCuts	Opt value, %	Sol time, s	Root node gap, %	nNodes	nCuts	Opt value, %	Sol time, s	Root node gap, %	nNodes	nCuts	Opt value, %	Sol time, s	Root node gap, %	nNodes	nCuts
1	1	30.54	1.9	0	0	1,183	33.29	7.7	0	0	1,173	36.46	14.9	0	0	1,173	37.66	20.3	0	0	1,200
	5	67.08	3.4	0	0	2,408	79.57	29.9	0	5	3,247	82.86	92.0	2.55	37	4,092	90.46	119.3	0.32	9	2,712
	10	87.98	4.1	0.66	5	2,362	93.47	25.3	1.18	22	2,478	94.90	49.5	0.22	34	2,696	98.82	90.7	0.15	25	1,952
	15	95.01	4.1	0	7	2,877	98.89	21.2	0	0	1,938	99.24	44.6	0.07	7	2,019	100.00	72.2	0	0	1,392
	20	98.41	4.5	0.34	100	2,569	99.82	17.8	0.02	5	1,680	99.97	35.1	0.01	34	1,444	100.00	70.8	0	0	1,295
	25	99.79	6.1	0.12	274	3,261	99.99	16.4	0.01	6	1,284	100.00	31.4	0	5	1,234	100.00	72.3	0	0	1,297
	30	100.00	4.7	0	98	2,340	100.00	15.1	0	0	1,215	100.00	38.6	0	0	1,283	100.00	72.0	0	0	1,297
	35	100.00	2.9	0	2	1,344	100.00	13.9	0	0	1,184	100.00	34.4	0	0	1,283	100.00	74.2	0	0	1,297
2	1	30.54	3.2	0	0	2,380	33.29	12.5	0	0	2,384	36.46	22.5	0	0	2,375	37.66	43.9	0	0	2,518
	5	71.05	6.5	0	1	4,365	81.12	48.7	0.08	2	5,000	84.14	142.7	0.4	52	6,150	91.61	250.2	0	14	4,783
	10	88.08	6.6	1.05	15	4,610	94.05	46.8	0.64	37	4,515	95.54	97.5	0.25	23	4,178	98.99	183.5	0	11	3,432
	15	95.27	6.8	0.16	9	4,683	98.90	34.1	0	7	3,098	99.32	93.2	0.03	9	3,289	100.00	142.7	0	0	2,506
	20	98.57	10.7	0.47	104	5,022	99.86	39.1	0.02	18	3,275	99.97	81.9	0.02	54	3,152	100.00	143.7	0	0	2,524
	25	99.85	9.7	0.08	276	3,399	99.99	34.4	0.01	155	2,590	100.00	78.3	0	0	2,636	100.00	149.8	0	0	2,443
	30	100.00	8.2	0	118	3,528	100.00	34.0	0	1	2,694	100.00	71.8	0	0	2,365	100.00	142.5	0	0	2,443
	35	100.00	5.5	0	5	2,963	100.00	30.0	0	0	2,352	100.00	73.3	0	0	2,365	100.00	150.8	0	0	2,443
3	1	30.65	5.0	0	0	3,592	33.41	21.7	0	0	3,617	36.58	50.4	0	0	3,712	38.78	70.7	0	0	3,775
	5	73.76	9.5	0	0	5,843	82.19	70.0	0.71	9	6,383	85.00	212.4	0.94	51	8,671	92.19	343.5	0.16	9	7,093
	10	89.04	10.6	1.01	11	6,603	94.83	74.1	0.28	25	6,292	96.34	137.6	0.15	21	5,357	99.07	277.7	0.09	21	4,777
	15	95.77	11.5	0.36	50	6,161	98.92	65.0	0	6	5,398	99.40	145.1	0.02	7	5,241	100.00	233.6	0	1	3,779
	20	98.79	17.9	0.41	156	7,380	99.87	62.7	0.02	23	4,494	99.98	116.2	0.02	80	4,011	100.00	237.7	0	0	3,685
	25	99.89	12.2	0.06	90	5,238	100.00	63.5	0	335	4,076	100.00	117.6	0	0	3,873	100.00	238.0	0	0	3,687
	30	100.00	9.8	0	5	4,655	100.00	55.9	0	1	3,722	100.00	121.1	0	0	3,595	100.00	237.8	0	0	3,687
	35	100.00	9.2	0	1	4,603	100.00	55.9	0	0	3,629	100.00	119.7	0	0	3,590	100.00	236.3	0	0	3,687
4	1	31.56	6.1	0	0	4,783	33.69	31.9	0	0	4,855	36.89	51.8	0	0	5,008	39.63	106.1	0	0	5,110
	5	75.88	12.8	0	1	7,704	83.12	94.3	0.03	9	8,141	86.00	264.7	0.26	61	10,653	92.90	483.8	0.1	11	9,605
	10	90.11	13.9	0.64	17	7,619	95.29	88.2	0.32	9	8,325	96.77	204.3	0.13	3	7,676	99.12	384.9	0.04	16	6,029
	15	96.10	18.9	0.52	99	7,945	98.97	88.7	0.06	21	7,765	99.45	198.2	0.06	19	6,177	100.00	327.5	0	1	4,966
	20	98.95	19.1	0.38	163	7,595	99.89	71.5	0.02	16	5,394	99.98	175.1	0.02	152	5,027	100.00	326.1	0	0	4,866
	25	99.91	16.2	0.05	129	6,069	100.00	72.3	0	9	4,777	100.00	158.9	0	4	5,235	100.00	353.3	0	0	4,868
	30	100.00	16.8	0	123	6,143	100.00	69.9	0	0	4,882	100.00	156.3	0	0	4,743	100.00	318.9	0	0	4,868
	35	100.00	9.6	0	0	5,089	100.00	68.2	0	0	4,862	100.00	157.9	0	0	4,738	100.00	332.9	0	0	4,868
5	1	32.54	8.6	0	0	5,993	33.91	34.9	0	0	6,108	37.13	80.3	0	0	6,378	40.47	146.4	0	0	6,682
	5	77.53	14.6	0	0	8,715	83.81	115.0	0.26	19	10,447	86.83	299.8	0.22	43	11,714	93.43	574.1	0.09	16	10,169
	10	90.93	17.7	0.46	11	9,137	95.67	113.0	0.32	7	9,913	97.11	238.2	0.1	7	8,246	99.24	492.9	0.04	6	7,380
	15	96.56	23.0	0.47	59	9,433	99.00	106.5	0.12	42	8,294	99.52	222.4	0	23	7,396	100.00	418.1	0	0	6,165
	20	99.05	33.1	0.36	270	8,884	99.90	94.1	0.02	14	6,502	99.98	208.6	0.01	107	6,221	100.00	488.7	0	0	6,010
	25	99.93	18.0	0.05	108	7,245	100.00	97.6	0	61	6,285	100.00	200.6	0	0	6,011	100.00	435.9	0	0	6,010
	30	100.00	19.0	0	41	7,701	100.00	82.7	0	0	6,027	100.00	192.9	0	0	5,953	100.00	463.5	0	0	6,010
	35	100.00	11.2	0	0	6,280	100.00	82.1	0	0	6,027	100.00	186.1	0	0	5,953	100.00	428.9	0	0	6,010

Table 6. (Continued)

		0% driver tolerance					10% driver tolerance					20% driver tolerance					50% driver tolerance				
Number of vehicle types	p	Opt value, %	Sol time, s	Root node gap, %	nNodes	nCuts	Opt value, %	Sol time, s	Root node gap, %	nNodes	nCuts	Opt value, %	Sol time, s	Root node gap, %	nNodes	nCuts	Opt value, %	Sol time, s	Root node gap, %	nNodes	nCuts
6	1	33.44	10.0	0	0	7,228	34.20	43.2	0	0	7,422	37.56	106.2	0	0	7,891	41.38	208.6	0	0	8,339
	5	78.77	18.6	0	0	10,354	84.76	149.1	0.23	19	12,171	87.53	390.2	0.28	80	14,545	93.81	694.3	0.1	17	12,087
	10	91.56	26.3	0.36	37	11,116	95.98	118.3	0.46	12	10,485	97.35	275.7	0.03	8	9,576	99.33	593.6	0	13	9,072
	15	96.89	30.1	0.44	63	10,577	99.10	127.5	0.11	47	9,566	99.58	247.8	0	9	8,637	100.00	538.1	0	0	7,125
	20	99.14	34.3	0.34	150	10,924	99.91	120.2	0.02	20	7,921	99.98	260.5	0.01	130	7,399	100.00	556.1	0	0	7,074
	25	99.94	31.6	0.04	290	10,321	100.00	108.1	0	10	7,371	100.00	223.3	0	7	7,184	100.00	517.0	0	0	7,074
	30	100.00	20.5	0	37	8,735	100.00	99.4	0	0	7,123	100.00	221.9	0	0	7,122	100.00	503.8	0	0	7,074
	35	100.00	15.8	0	1	7,498	100.00	96.5	0	0	7,123	100.00	256.7	0	0	7,122	100.00	491.0	0	0	7,074
7	1	34.08	11.9	0	0	8,498	34.73	54.7	0	0	8,782	37.99	119.4	0	0	9,501	42.13	287.6	0	0	10,265
	5	79.91	21.6	0	0	12,186	85.57	180.1	0.51	17	13,729	88.17	496.3	0.26	114	17,845	94.07	823.5	0.12	17	14,424
	10	92.21	27.2	0.25	13	12,542	96.32	148.4	0.34	14	11,258	97.53	319.0	0.06	11	10,846	99.39	680.2	0	24	9,549
	15	97.15	48.1	0.42	152	12,374	99.18	141.1	0.1	31	10,539	99.63	301.8	0	0	9,780	100.00	668.0	0	0	8,292
	20	99.23	50.6	0.31	418	10,563	99.92	145.4	0.02	22	9,065	99.99	299.9	0.01	122	8,594	100.00	639.2	0	0	8,241
	25	99.94	34.4	0.03	257	9,954	100.00	122.3	0	27	8,478	100.00	315.9	0	0	8,320	100.00	644.0	0	0	8,241
	30	100.00	23.2	0	22	9,180	100.00	125.7	0	0	8,290	100.00	276.0	0	0	8,289	100.00	623.5	0	0	8,241
	35	100.00	17.6	0	3	8,671	100.00	112.4	0	0	8,290	100.00	278.1	0	0	8,289	100.00	608.6	0	0	8,241
8	1	34.88	15.3	0	0	9,766	35.56	63.5	0	0	10,092	38.60	157.1	0	0	11,120	43.01	367.1	0	0	12,051
	5	80.84	25.8	0	1	13,565	86.18	174.5	0.41	19	14,778	88.65	479.9	0.96	88	17,542	94.47	1007.2	0.09	21	16,476
	10	92.71	29.3	0.23	14	13,144	96.60	162.0	0.18	13	12,504	97.71	390.3	0.21	16	11,967	99.44	765.5	0	7	10,829
	15	97.35	51.0	0.43	162	13,216	99.25	166.6	0.1	26	11,685	99.66	355.1	0	0	10,923	100.00	681.7	0	0	9,459
	20	99.30	41.7	0.29	204	12,144	99.93	144.6	0.02	7	9,970	99.99	363.7	0.01	115	9,856	100.00	655.3	0	0	9,408
	25	99.95	33.4	0.03	184	10,896	100.00	149.3	0	8	9,843	100.00	361.8	0	0	9,487	100.00	672.2	0	0	9,408
	30	100.00	38.5	0	295	10,147	100.00	137.7	0	0	9,457	100.00	338.0	0	0	9,456	100.00	714.4	0	0	9,408
	35	100.00	21.7	0	1	9,830	100.00	134.5	0	0	9,457	100.00	333.9	0	0	9,456	100.00	633.1	0	0	18,457

Figure 6. Solution Times for Different Number of Vehicle Types



distribution of deviation distances. For the considered setting, Figure 4(a) shows that 9.53% of the total vehicle flow is missed, 13.86% of the total vehicle flow drive on their shortest paths (shown in the first bar in Figure 5), and 76.61% of the total vehicle flow deviate (shown in bars 2–11 in Figure 5). The deviation peaks at 5%–10% deviation and gradually decreases until 50% tolerance level. The average devDist in the figure is 12.79%. When all the instances in Table 5 are considered, the average devDist for $\hat{\lambda} = 10\%$, $\hat{\lambda} = 20\%$, and $\hat{\lambda} = 50\%$ are 3.1%, 5.8%, and 10.6%, respectively.

Observe that there are three critical parameters in the RSLP-R that affect the performance: (1) the deviation tolerance of the drivers, (2) the number of demands, and (3) the size of the network. In the CA network, we increased the deviation tolerance to 50% and observed that the model adapted to increasing deviations. In the following experiment, we concentrate on increasing the number of demands, and finally, we test our model against increasing network sizes in random graphs.

In the CA network, there are 1,167 OD pairs. The results presented in Table 5 are obtained assuming a single vehicle type traveling between each OD pair. In the following experiment, we add a new parameter: the number of vehicle types, and each OD pair is assigned the same number of vehicle types. Therefore, having eight vehicle types in an experiment refers to 9,336 distinct demands traveling between 1,167 OD pairs. The ranges of vehicles change between 100 and 275 kilometers by increments of 25. In other words, when we have eight vehicle types running between the same OD pair, their ranges are 100, 125, . . . , 275 kilometers. In this regard, Table 6 presents results for the CA network for a different number of vehicle types. We again observe very small root node gaps. Figure 6 depicts the average solution times of the 32 instances corresponding to each vehicle type and shows that the solution times sublinearly increase by the number of vehicle types.

We also investigate the effect of increasing the number of OD pairs on solution times. For this purpose, we introduce two new sets of OD pairs. Recall that the 1,167 OD pair nodes in the previous experiments represent those urban centers with a population of 50,000 or more. This corresponds to approximately 5.71 million vehicles per year. In the first (and second, respectively) new set, we consider those urban centers with 30,000 (and 20,000, respectively) or more in population and 30 kilometers apart, corresponding to 1,874 (and 3,121, respectively) OD pairs and more than 5.91 million (and 6.11 million, respectively) vehicles per year. The results are reported in columns (3)–(20) of Table 7. All instances are solved in less than eight minutes. Apart from these two new OD pair sets, we also test unit vehicle flows between each OD pair. In our original experiments with 1,167 OD pairs, the vehicle flows are highly unbalanced, changing between 14.51 and 1,235,110 vehicles per year. For both balanced and unbalanced instances, the solution times and the root node gaps are consistently small, showing the strength of our solution methodology.

4.2. Random Networks

Random networks are generated to test the computational efficiency of the proposed B&C-1 algorithm against increasing sizes of networks. For this purpose, four random graphs, details of which are presented in Table 2, are generated. For each network, 1,000, 2,000, and 4,000 random OD pairs are selected. For each graph and OD pair count, we test our algorithm for 10, 20, 30, 50, and 100 stations. We conduct each experiment for 0%, 10%, and 20% deviation tolerance. We consider a vehicle with a range of 100 kilometers. The results are presented in Table 8. Three leftmost columns show the parameter settings. The "Best lbd, %" column shows the best feasible solution at termination. "Opt Gap, %" is the gap between the best upper and lower bounds at termination. "Sol time, s," "Root node gap, %," "nNodes," and "nCuts" columns are as previously defined.

Observe that except for 17 instances, the algorithm was able to find the optimal solution within the one-hour time limit. Average solution times are generally higher than those for the CA network. Note that even with large gaps at the root node, the algorithm still performs fairly fast in finding an optimal solution. According to our results, the algorithm performs well until the bottleneck is hit in instances with 1,000 nodes and 4,000 OD pairs.

5. Conclusions

In this study, we have proposed a natural formulation for the RSLP-R based on the notion of length-bounded node cuts and analyzed its polyhedral properties. We proposed a B&C algorithm as an exact solution method. For separating integer solutions, we devised a polynomial-time algorithm. For fractional solutions, we developed an integer programming model and

Table 7. Results for the CA Instances with Different OD Pair Numbers

		Number of OD pairs = 1,167 (total flow = 5,718,328 vehicles/year)					Number of OD pairs = 1,874 (total flow = 5,912,295 vehicles/year)					Number of OD pairs = 3,121 (total flow = 6,115,458 vehicles/year)					Number of ODpairs = 1,167 (unit flow for each OD pair)				
p	Tol, %	Opt value (1,000 vehicles/ year)	Sol time, s	Root node gap, %	nNodes	nCuts	Opt value (1,000 vehicles/ year)	Sol time, s	Root node gap, %	nNodes	nCuts	Opt value (1,000 vehicles/ year)	Sol time, s	Root node gap, %	nNodes	nCuts	Opt value	Sol time, s	Root node gap, %	nNodes	nCuts
1	1	1,747	1.9	0	0	1,183	1,748	2.7	0	0	1,893	1,770	4.2	0	0	3,138	77	1.8	0	0	1,169
	1.1	1,903	7.7	0	0	1,173	1,928	10.9	0	0	1,885	1,972	19.6	0	0	3,138	89	7.3	0	0	1,170
	1.2	2,085	14.9	0	0	1,173	2,110	23.7	0	0	1,885	2,145	29.4	0	0	3,139	98	12.1	0	0	1,170
	1.5	2,154	20.3	0	0	1,200	2,183	44.5	0	0	1,918	2,226	55.1	0	0	3,196	114	20.1	0	0	1,181
5	1	3,836	3.4	0	0	2,408	3,877	4.9	0	1	3,707	3,946	8.8	0	0	6,041	387	3.7	16.54	21	2,834
	1.1	4,550	29.9	0.38	5	3,247	4,620	53.8	0.7	5	5,223	4,719	98.2	1.42	7	8,857	524	32.2	8.68	116	3,367
	1.2	4,738	92.0	4.82	34	3,818	4,812	125.6	0.95	25	5,522	4,919	234.4	1.60	43	9,771	625	83.1	7.73	132	3,839
	1.5	5,173	119.3	0.50	9	2,712	5,279	285.7	0.71	33	6,586	5,381	467.1	1.13	77	9,902	794	247.8	6.52	125	6,431
10	1	5,031	4.1	0.78	5	2,362	5,145	5.4	0.59	5	3,489	5,267	9.0	0.63	5	5,991	699	4.0	1.96	21	3,015
	1.1	5,345	25.3	1.18	22	2,478	5,472	41.7	0.43	11	3,643	5,617	91.8	0.38	24	7,200	913	24.0	2.37	28	2,349
	1.2	5,427	49.5	0.39	25	2,630	5,576	93.5	0.16	56	4,092	5,715	166.3	0.69	61	7,152	1,022	38.3	0.44	25	1,937
	1.5	5,651	90.7	0.25	25	1,952	5,811	180.0	0.28	41	3,518	5,962	351.9	0.29	13	6,681	1,132	88.1	0.39	5	1,936
15	1	5,433	4.1	0.06	7	2,877	5,574	5.9	0.1	0	4,263	5,707	11.2	0	1	7,002	939	4.2	2.24	7	3,000
	1.1	5,655	21.2	0.01	4	1,938	5,800	34.2	0	1	2,829	5,952	72.4	0	3	5,347	1,116	18.6	0.09	0	1,600
	1.2	5,675	44.6	0.08	7	2,019	5,833	92.9	0.17	21	3,590	6,005	126.9	0.07	12	5,051	1,146	33.6	0.09	5	1,396
	1.5	5,718	72.2	0	0	1,392	5,903	208.1	0.05	19	3,751	6,092	257.1	0.05	2	5,098	1,167	70.8	0	0	1,348
20	1	5,627	4.5	0.48	72	2,556	5,775	7.7	0.42	71	4,647	5,915	11.0	0.47	27	6,684	1,101	4.8	1.35	50	3,240
	1.1	5,708	17.8	0.02	5	1,680	5,871	34.0	0.03	12	2,841	6,041	62.2	0.02	3	4,745	1,154	16.9	0.11	6	1,396
	1.2	5,717	35.1	0.02	45	1,446	5,897	80.8	0.07	32	3,120	6,080	133.9	0.03	2	4,643	1,162	33.7	0.17	14	1,421
	1.5	5,718	70.8	0	0	1,295	5,912	122.0	0	1	2,300	6,114	203.7	0	1	3,576	1,167	73.9	0	0	1,295
25	1	5,707	6.1	0.12	247	3,217	5,875	8.7	0.19	227	4,072	6,025	13.3	0.11	98	6,489	1,147	4.7	0.93	301	2,444
	1.1	5,718	16.4	0.01	10	1,284	5,902	29.9	0.06	71	2,371	6,086	57.2	0.03	13	4,272	1,166	15.3	0.09	10	1,248
	1.2	5,718	31.4	0	5	1,234	5,910	73.3	0.01	54	2,576	6,108	113.5	0.02	30	4,092	1,167	32.4	0	5	1,234
	1.5	5,718	72.3	0	0	1,297	5,912	117.1	0	0	2,149	6,115	210.7	0	0	3,475	1,167	71.3	0	0	1,297
30	1	5,718	4.7	0	16	2,456	5,902	7.7	0.09	495	3,473	6,072	12.3	0.15	115	6,235	1,167	3.6	0	0	1,897
	1.1	5,718	15.1	0	0	1,215	5,912	29.6	0	20	2,317	6,108	52.7	0.01	6	4,160	1,167	15.5	0	0	1,215
	1.2	5,718	38.6	0	0	1,283	5,912	75.0	0	41	2,504	6,114	105.8	0	29	3,773	1,167	34.3	0	0	1,283
	1.5	5,718	72.0	0	0	1,297	5,912	119.8	0	0	2,138	6,115	201.5	0	0	3,373	1,167	72.6	0	0	1,297
35	1	5,718	2.9	0	2	1,344	5,911	5.3	0	63	2,827	6,102	12.4	0.13	226	5,720	1,167	3.1	0	2	1,344
	1.1	5,718	13.9	0	0	1,184	5,912	27.6	0	0	2,061	6,114	53.7	0	12	3,990	1,167	15.7	0	0	1,184
	1.2	5,718	34.4	0	0	1,283	5,912	63.5	0	3	2,221	6,115	107.6	0	14	3,746	1,167	32.7	0	0	1,283
	1.5	5,718	74.2	0	0	1,297	5,912	118.2	0	0	2,138	6,115	217.1	0	0	3,288	1,167	72.4	0	0	1,297

Table 8. Results for Randomly Generated Graphs

			0% driver tolerance						10% driver tolerance						20% driver tolerance					
Network	Number of OD pairs	p	Best lbd, %	Opt gap, %	Sol time, s	Root node gap, %	nNodes	nCuts	Best lbd, %	Opt gap, %	Sol time, s	Root node gap, %	nNodes	nCuts	Best lbd, %	Opt gap, %	Sol time, s	Root node gap, %	nNodes	nCuts
G-250	1,000	10	14.91	0	2.0	11.78	164	1,781	23.15	0	1.7	6.63	46	1,608	27.35	0	3.5	6.68	134	1,682
		20	29.69	0	3.3	7.58	510	1,752	43.56	0	1.9	1.57	41	1,643	50.54	0	3.5	0.52	19	1,577
		30	43.51	0	2.7	4.47	298	1,728	60.82	0	1.8	0.12	4	1,610	67.4	0	3.4	0.59	38	1,590
		50	67.41	0	1.8	0.47	69	1,743	85.96	0	2.1	0.25	14	1,467	91.87	0	3.1	0.06	4	1,324
		100	94.71	0	1.3	0.00	1	1,320	100	0	1.7	0.00	0	1,152	100	0	3.1	0.00	0	1,075
	2,000	10	15.06	0	4.4	10.09	254	3,315	23.69	0	3.5	4.49	25	2,982	27.74	0	5.4	4.71	75	2,886
		20	29.54	0	6.8	6.40	435	3,360	44.89	0	3.4	1.10	23	3,098	50.11	0	5.6	0.68	17	2,890
		30	42.75	0	10.1	5.19	848	3,449	61.63	0	3.6	0.56	24	3,153	68.02	0	6.1	0.02	3	3,071
		50	65.82	0	3.7	0.99	30	3,229	87.18	0	3.5	0.10	6	2,891	92.09	0	6.3	0.14	5	2,665
		100	91.91	0	3.5	0.24	58	2,615	100	0	2.8	0.00	0	2,113	100	0	4.0	0.00	0	2,097
	4,000	10	14.72	0	15.0	12.33	372	6,604	24.37	0	7.2	2.92	12	6,101	29.02	0	10.2	2.23	27	5,614
		20	29.86	0	15.9	4.94	415	6,569	45.48	0	8.1	0.78	25	6,071	51.6	0	11.9	0.72	17	6,004
		30	43.92	0	9.5	1.24	40	6,622	61.95	0	8.1	0.05	3	6,103	68.12	0	11.7	0.40	10	5,822
		50	64.58	0	14.2	1.88	227	6,565	86.6	0	7.0	0.00	1	5,236	91.34	0	11.4	0.00	1	4,997
		100	91.43	0	9.3	0.41	80	5,436	100	0	6.3	0.00	0	4,231	100	0	8.9	0.00	0	4,224
G-500	1,000	10	10.91	0	3.4	9.64	184	1,676	16.27	0	5.7	8.56	618	1,609	18	0	14.1	11.70	2,884	1,616
		20	20.99	0	3.2	4.68	238	1,593	30.33	0	4.7	3.26	171	1,559	33.58	0	10.1	2.69	247	1,586
		30	29.38	0	4.3	3.77	520	1,634	42.13	0	5.2	1.94	132	1,590	46.73	0	10.2	0.74	37	1,570
		50	44.8	0	5.7	3.40	869	1,733	61.27	0	5.1	0.84	28	1,619	66.17	0	10.9	0.96	210	1,570
		100	73.59	0	4.4	0.96	165	1,769	93.02	0	4.9	0.11	8	1,346	96.47	0	11.6	0.00	1	1,273
	2,000	10	9.67	0	9.7	16.99	596	3,204	16.01	0	10.5	10.59	295	3,061	18.34	0	19.9	8.57	470	3,164
		20	19.03	0	8.2	7.24	373	3,162	30.23	0	9.1	1.67	55	3,006	34.47	0	17.7	2.19	87	3,048
		30	27.03	0	15.2	4.22	885	3,241	41.8	0	10.5	1.74	159	3,017	47.33	0	20.9	1.92	159	3,053
		50	41.13	0	51.9	3.88	5,730	3,426	60.83	0	11.8	1.18	291	2,949	67.48	0	18.1	0.32	43	2,857
		100	69.26	0	48.3	2.22	4,124	3,393	90.72	0	10.7	0.11	18	2,662	94.72	0	18.9	0.04	17	2,477
	4,000	10	10.38	0	11.0	9.55	104	6,181	17.13	0	13.9	2.18	34	5,706	19.74	0	35.3	2.06	33	5,849
		20	18.91	0	40.5	6.77	1,014	6,528	29.99	0	20.1	1.89	86	6,314	34.11	0	44.0	2.51	387	5,930
		30	25.82	0	252.4	6.81	9,115	6,755	40.45	0	22.6	2.39	415	6,026	45.37	0	73.9	2.93	1767	6,025
		50	38.94	0	3,604.6[a]	N/A	116,059	6,936	58.69	0	40.0	1.65	963	6,188	64.22	0	63.7	1.53	1323	5,794
		100	67.19	0	160.5	2.18	5,628	6,601	89.6	0	17.2	0.04	5	5,200	93.67	0	36.8	0.06	21	4,737
G-750	1,000	10	7	0	7.7	19.28	511	1,844	10.85	0	10.7	14.98	485	1,934	13.36	0	24.9	12.92	1,588	1,829
		20	13.99	0	8.7	10.35	589	1,854	21.34	0	11.7	7.62	402	1,845	24.93	0	32.5	8.74	1,967	1,951
		30	20.65	0	8.2	6.04	476	1,886	30.13	0	15.8	5.94	1,710	1,832	35.06	0	47.4	6.20	4,387	2,052
		50	32.14	0	31.4	6.11	5,978	1,951	45.89	0	25.1	4.04	3,450	1,994	53.07	0	27.7	2.49	1,024	1,949
		100	57.27	0	18.6	2.14	2,544	1,978	78.31	0	10.1	0.45	63	1,771	84.85	0	23.3	0.58	300	1,619
	2,000	10	6.64	0	13.4	18.66	544	3,468	10.98	0	22.5	14.56	790	3,559	13.8	0	48.4	13.82	1,394	3,421
		20	12.89	0	26.1	12.41	1,448	3,616	20.51	0	31.1	6.98	1,745	3,612	25.46	0	69.5	5.80	3,650	3,590
		30	18.6	0	47.2	8.34	3,466	3,592	28.59	0	164.7	7.00	19,547	3,748	35.06	0	127.5	4.62	10,518	3,623
		50	29.41	0	82.8	5.80	6,322	3,743	44.23	0	47.1	3.31	3,606	3,473	51.76	0	62.1	2.12	2,432	3,371
		100	52.56	0	180.6	2.93	15,147	3,719	73.41	0	20.7	0.87	527	3,312	81.52	0	40.2	0.25	26	3,082
	4,000	10	5.9	0	93.1	23.48	1,750	7,316	10.34	0	89.9	16.24	3,599	6,815	12.81	0	152.4	16.59	2,785	7,209

Table 8. (Continued)

			0% driver tolerance						10% driver tolerance						20% driver tolerance					
Network	Number of OD pairs	p	Best lbd, %	Opt gap, %	Sol time, s	Root node gap, %	nNodes	nCuts	Best lbd, %	Opt gap, %	Sol time, s	Root node gap, %	nNodes	nCuts	Best lbd, %	Opt gap, %	Sol time, s	Root node gap, %	nNodes	nCuts
		20	11.41	0	923.9	14.08	25,600	7,116	20.56	0	42.0	3.94	163	7,016	24.93	0	103.2	4.59	752	6,862
		30	16.59	0.03	3,609.3^a	N/A	80,559	7,295	28.34	0	171.0	5.47	3,599	7,292	34.31	0	113.9	2.63	507	7,024
		50	26.79	0.03	3,609.0^a	N/A	65,399	7,274	42.6	0	1,764.8	4.62	53,543	7,162	49.84	0	1,073.1	2.88	37,632	7,084
		100	50.04	0.01	3,609.7^a	N/A	75,809	7,392	73.32	0	45.4	0.49	298	6,318	80.21	0	84.4	0.16	50	5,818
G-1000	1,000	10	4.95	0	13.3	30.32	748	1,958	7.51	0	52.9	45.00	5,498	2,392	9.03	0	112.1	36.04	12,696	2,465
		20	9.82	0	21.5	14.79	2,123	1,966	15.53	0	39.5	14.74	4,002	2,225	18.46	0	229.3	18.90	30,193	2,376
		30	14.71	0	30.9	10.15	3,244	2,125	23.93	0	23.3	9.68	1,552	2,119	28.37	0	85.7	9.01	6,824	2,284
		50	24.08	0	47.5	6.55	6,003	2,030	38.64	0	36.9	3.64	3,251	2,103	44.83	0	208.9	4.66	20,513	2,303
		100	44.36	0	655.7	4.56	113,827	2,089	67.18	0	27.2	1.15	2,634	2,073	74.99	0	186.2	1.67	33,113	2,129
	2,000	10	4.58	0	24.9	28.94	610	3,696	6.91	0	117.4	45.38	6,214	4,175	8.02	0	843.9	50.80	36,722	4,884
		20	8.74	0	164.3	20.31	8,980	3,860	14.95	0	111.1	15.93	4,774	4,055	17.96	0	443.2	16.06	22,291	4,313
		30	13.22	0	272.6	13.82	14,575	4,076	22.14	0	464.6	10.55	28,069	3,903	27.3	0	245.7	7.02	10,997	4,087
		50	21.9	0	1,858.6	8.74	123,478	3,756	35.16	0.01	3,612.3^a	N/A	169,123	4,071	41.93	0	2,862.0	6.11	17,1187	4,297
		100	40.84	0.02	3,610.4^a	N/A	145,421	4,078	62.1	0	306.6	1.93	19,311	3,958	69.67	0	321.8	1.67	21,154	3,760
	4,000	10	3.63	0	1,120.0	46.67	14,060	8,392	5.73	0	3,184.2	49.99	65,238	8680	6.71	0.26	3,632.2^a	N/A	59,069	8,489
		20	7.58	0	3,349.8	26.70	58,719	7,949	12.62	0.05	3,622.7^a	N/A	59,302	8,388	15.66	0.10	3,645.2^a	N/A	57,580	8,088
		30	11.16	0.08	3,616.7^a	N/A	49,983	7,835	19.95	0.02	3,623.7^a	N/A	80,167	7,779	24.63	0.05	3,645.3^a	N/A	60,854	7,778
		50	18.28	0.11	3,616.3^a	N/A	47,151	7,853	33.42	0.01	3,622.5^a	N/A	86,946	7,570	40.51	0.01	3,645.2^a	N/A	66,932	7,730
		100	37.04	0.06	3,616.1^a	N/A	48,154	7,807	58.96	0.01	3,622.9^a	N/A	87,638	7,256	67.97	0	165.8	1.00	1,828	7,278

[a]Instances terminated because of the time limit.

made use of a classical minimum cut algorithm as an efficient heuristic. Extensive computational studies showed that the solution times and the size of the solvable instances are improved significantly with respect to previously reported results by Yıldız et al. (2016). We can address practically sized problem instances, which could only be solved previously by a B&P algorithm in a three-hour time frame, in under two minutes. Our approach can also address multiple vehicle types and possible nonsimple path occurrences in the driver routes.

Our approach can easily be adapted to handle different driver tolerances. Rather than assuming a distance tolerance on the path, a bound on the number of refueling stops might be considered for the drivers. This special case of our problem can be solved by assuming unit length arcs in the transformed network and solving the RSLP-R on this transformed network.

For future research directions, capacities of the refueling stations can be accommodated into the problem, which would bring more realism. Considering uncertainties inherent in the problem, such as the vehicle flows between OD pairs, can also help in modeling the real world more closely.

Acknowledgments
The authors thank the associate editor and two referees for their constructive comments that helped improve the quality of this paper.

References
Arslan O, Karaşan OE (2016) A Benders decomposition approach for the charging station location problem with plug-in hybrid electric vehicles. *Transportation Res. Part B: Methodological* 93: 670–695.

Arslan O, Yıldız B, Karaşan OE (2014) Impacts of battery characteristics, driver preferences and road network features on travel costs of a plug-in hybrid electric vehicle (PHEV) for long-distance trips. *Energy Policy* 74:168–178.

Baier G, Erlebach T, Hall A, Köhler E, Schilling H, Skutella M (2006) Length-bounded cuts and flows. Bugliesi M, Preneel B, Sassone V, Wegener I, eds. *Internat. Colloquium Automata, Languages, Programming* (Springer, Berlin, Heidelberg), 679–690.

Capar I, Kuby M, Leon VJ, Tsai YJ (2013) An arc-cover path-cover formulation and strategic analysis of alternative-fuel station locations. *Eur. J. Oper. Res.* 227(1):142–151.

Chen S, Ljubić I, Raghavan S (2010) The regenerator location problem. *Networks* 55(3):205–220.

Church R, ReVelle CS (1974) The maximal covering location problem. *Papers Regional Sci. Assoc.* 32(1):101–118.

Energy Information Administration (2017a) Alternative fuel vehicle data. Accessed January 17, 2017, http://www.eia.gov/renewable/afv/users.php?fs=a.

Energy Information Administration (2017b) Annual energy outlook. Accessed May 18, 2014, http://www.eia.gov/outlooks/aeo/.

European Commission (2013) Clean power for transport. Accessed January 17, 2017, http://europa.eu/rapid/press-_release_MEMO-_13-_24_en.htm.

European Union (2009) Renewable energy directive. Accessed January 16, 2017, http://www.eur-_lex.europa.eu/legal-_content/EN/TXT/PDF/?uri=CELEX:32009L0028.

Hodgson MJ (1990) A flow-capturing location-allocation model. *Geographical Anal.* 22(3):270–279.

IBM (2014) ILOG CPLEX optimization studio. Accessed February 13, 2017, http://www.cplex.com.

Kim JG, Kuby M (2012) The deviation-flow refueling location model for optimizing a network of refueling stations. *Internat. J. Hydrogen Energy* 37(6):5406–5420.

Kim JG, Kuby M (2013) A network transformation heuristic approach for the deviation flow refueling location model. *Comput. Oper. Res.* 40(4):1122–1131.

Kuby M, Lim S (2005) The flow-refueling location problem for alternative-fuel vehicles. *Socio-Econom. Planning Sci.* 39(2): 125–145.

Li X, Aneja Y (2017) Regenerator location problem: Polyhedral study and effective branch-and-cut algorithms. *Eur. J. Oper. Res.* 257(1): 25–40.

Lim S, Kuby M (2010) Heuristic algorithms for siting alternative-fuel stations using the flow-refueling location model. *Eur. J. Oper. Res.* 204(1):51–61.

Lovász L, Neumann-Lara V, Plummer M (1978) Mengerian theorems for paths of bounded length. *Periodica Mathematica Hungarica* 9(4):269–276.

Mahjoub AR, McCormick ST (2010) Max flow and min cut with bounded-length paths: Complexity, algorithms, and approximation. *Math. Programming* 124(1–2):271–284.

MirHassani SA, Ebrazi R (2013) A flexible reformulation of the refueling station location problem. *Transportation Sci.* 47(4): 617–628.

Sichi J, Kinable J, Michail D, Naveh B, Contributors (2017) JGraphT—Graph algorithms and data structures in Java. Accessed February 13, 2017, http://www.jgrapht.org.

PassMark Software (2017) CPU performance comparison. Accessed May 18, 2017, https://www.cpubenchmark.net/compare.php?cmp[]=2052\&cmp[]=1846.

Rahman Q, Bandyopadhyay S, Aneja Y (2015) Optimal regenerator placement in translucent optical networks. *Optical Switching Networking* 15:134–147.

Wang YW, Lin CC (2009) Locating road-vehicle refueling stations. *Transportation Res. Part E: Logist. Transportation Rev.* 45(5): 821–829.

Wang YW, Lin CC (2013) Locating multiple types of recharging stations for battery-powered electric vehicle transport. *Transportation Res. Part E: Logist. Transportation Rev.* 58:76–87.

Wang YW, Wang CR (2010) Locating passenger vehicle refueling stations. *Transportation Res. Part E: Logist. Transportation Rev.* 46(5):791–801.

Yetginer E, Karasan E (2003) Regenerator placement and traffic engineering with restoration in GMPLS networks. *Photonic Network Comm.* 6(2):139–149.

Yıldız B, Karaşan OE (2015) Regenerator location problem and survivable extensions: A hub covering location perspective. *Transportation Res. Part B: Methodological* 71:32–55.

Yıldız B, Arslan O, Karaşan OE (2016) A branch and price approach for routing and refueling station location model. *Eur. J. Oper. Res.* 248(3):815–826.