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

Okan Arslan
HEC Montréal and CIRRELT, 3000 chemin de la Côte-Sainte-Catherine, H3C 3J7 Montréal, Canada
okan.arslan@hec.ca

Oya Ekin Karaşan
Bilkent University, Department of Industrial Engineering, Bilkent, 06800 Ankara, Turkey
karasan@bilkent.edu.tr

A.Ridha Mahjoub
Université Paris Dauphine, PSL Research University, CNRS [7243], LAMSADE, 75016 Paris, France
ridha.mahjoub@lamsade.dauphine.fr

Hande Yaman
Bilkent University, Department of Industrial Engineering, Bilkent, 06800 Ankara, Turkey
hyaman@bilkent.edu.tr

Due to the limited range of alternative fuel vehicles (AFV) and the sparsity of the available alternative refueling stations (AFS), the 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 trade-offs between covered vehicle flow and deviation tolerance of the drivers and present insights on deviation characteristics of drivers in a case study in California.

1. Introduction

The purpose of this paper is to develop a branch-and-cut 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. Due to their limited supply levels, reserves are destined to deplete. Historically, 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 Comission 2013).

Due to the rather limited range of AFVs, availability of refueling stations in intercity transportation is a serious barrier on 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 since 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 (FRLP)*, 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. The 'flow refueling location problem' and the 'refueling station location problem' are used interchangeably in the literature to refer to the same problem (MirHassani and Ebrazi 2013, Yıldız, Arslan, and Karaşan 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 is 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. Due to 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, Ljubić, and Raghavan (2010) in the context of regenerator placement problem. The second work by Capar et al. (2013) refine the modeling logic using the fact that, in order 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, Arslan, and Karaşan (2016) present a branch-and-price algorithm to solve this problem. The computational gains of the branch-and-price 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, Ljubić, and Raghavan 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 regenerator placement problem show that branch-and-cut approaches outperform other exact solution methods, especially in large-scale instances (Rahman, Bandyopadhyay, and Aneja 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 branch-and-cut 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 that could only be solved previously by a branch-and-price 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 non-simple path occurrences in the routes. Using our methodology, we investigate the trade-offs 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 branch-and-cut 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), MirHassani and Ebrazi (2013) for further details.

Definition 1. The refueling station location problem with routing (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.

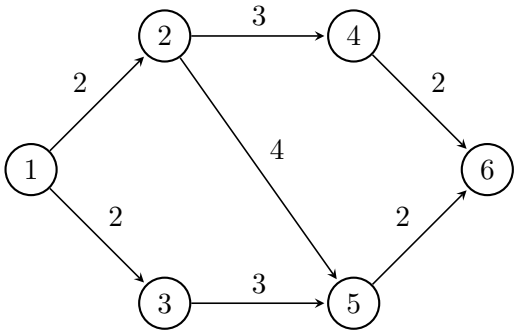
A node is called a 'dominated node' for a demand q if it does not appear on any path from o_q to d_q of length at most λ_q. Note that we can identify whether a node i is dominated by checking the shortest path distance between o_q and d_q using node i. Having $\delta_{o_q i} + \delta_{i d_q} > \lambda_q$ 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 where OD paths are not

fixed. For each $q \in Q$, consider graph $G_q = (N_q, A_q)$ with node set $N_q = \{s_q, t_q\} \cup \{i \in N : \delta_{o_q i} + \delta_{i d_q} \leq \lambda_q\}$ 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

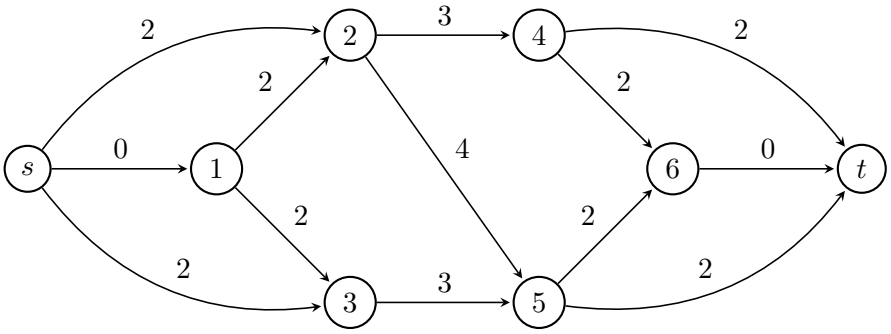
$$\begin{aligned} A_q^1 =& \{(s_q, j) : \delta_{o_q j} \leq r_q/2, j \in N_q \setminus \{s_q, t_q\}\}, \\ A_q^2 =& \{(i, t_q) : \delta_{i d_q} \leq r_q/2, i \in N_q \setminus \{s_q, t_q\}\}, \\ A_q^3 =& \{(i, j) : \delta_{ij} \leq r_q, i, j \in N_q \setminus \{s_q, t_q\}, i \neq j\}. \end{aligned}$$

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 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 the 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 node 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*.

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 $\mathcal{P}_q$.



(a) Graph representation of the road network



(b) Transformed graph

Figure 1 Length bounded cut example

Figure 1 (a) and (b) display 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 while 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 s-(1)-2-5-(6)-t 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.

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

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

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:

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

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

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

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

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

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 since 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.

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

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

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

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 $|N| \geq 2$. 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 Appendix.

PROPOSITION 1. *$conv(\mathcal{X})$ 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 $y_q \geq 0$ is facet defining for $conv(\mathcal{X})$.*

PROPOSITION 3. *For $i \in N$, the inequality $x_i \geq 0$ 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 $x_i \leq 1$ is facet defining for $conv(\mathcal{X})$ if and only if $p \geq 2$ and there exists a path $\pi^q \in \mathcal{P}'_q$ such that $|N(\pi^q) \cup \{i\}| \leq p$ 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 $x(N) \leq p$ 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 $y_q \leq x(S)$ 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 $|N(\pi^q) \cup N(\pi^{\hat{q}})| \leq p$.*

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* where all arcs have unit lengths is called the length-bounded minimum node-cut problem (Lovász, Neumann-Lara, and Plummer 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 1 if node $i \in N_q \setminus \{s_q, t_q\}$ is in the q-node-cut, and 0 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):

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

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

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

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

$$\pi_i \geq 0 \qquad \forall i \in N_q, \tag{13}$$

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

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 where we use this model for separation. However, this turns out to be computationally costly in most cases. For that 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 *mincut* heuristic. Note that, the minimum node-cuts we obtain by the heuristic are not necessarily minimal q-node-cuts since 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 branch-and-cut (B&C) algorithm to solve the *RSLP-R* since 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 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 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 on the other hand is designed to test if 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.

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

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 Java programming environment under Linux using Concert Technology. The computer has Intel Xeon E5-2630 v2 processor at 2.60 GHz and 96GB of RAM. The algorithms are implemented using callback classes. A time limit of one hour is set in all implementations.

Characteristics of network topologies considered in this study are detailed in Table 2. CA is a real-world representation of California road network with 339 nodes and 1234 arcs as shown in Figure 2 (Arslan, Yıldız, and Karaşan 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 1167 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 1000 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 vary between 1000 and 4000.

Table 2 Characteristics of instances

			node degree			OD pairs		
network	# nodes	# arcs	min	mean	max	min dist	mean dist	max dist
CA	339	1234	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	1284	2	5.14	20	30.05	142.95	366.72
G-750	750	1922	2	5.13	16	30.00	153.87	522.43
G-1000	1000	2580	2	5.16	22	30.00	160.98	458.86

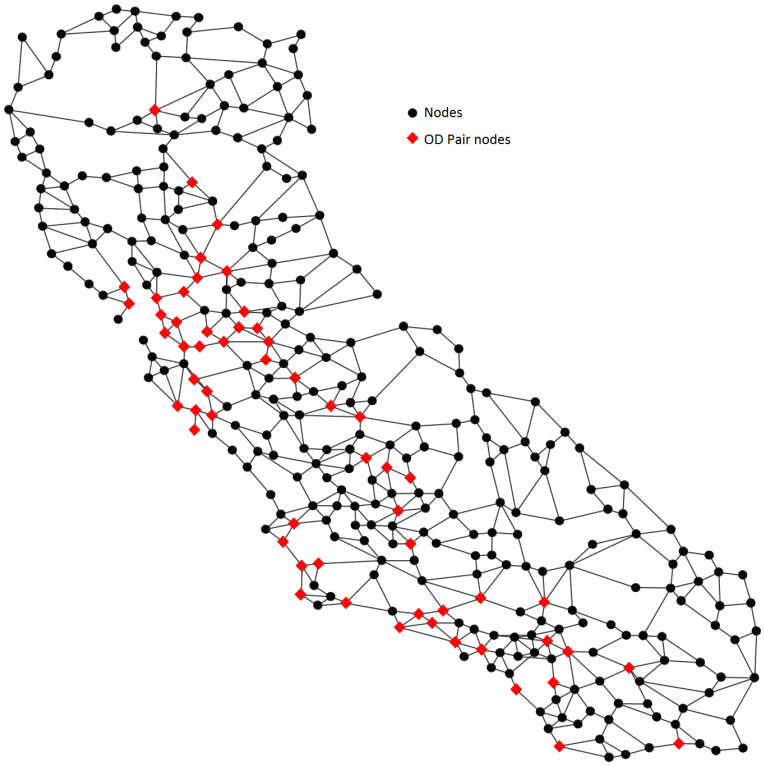


Figure 2 **California road network**

We mainly compare our results with the results of the branch-and-price algorithm by Yıldız, Arslan, and Karaşan (2016). Kim and Kuby (2013) present a heuristic algorithm based on network transformation, however since 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 branch-and-price algorithm is previously presented by Yıldız, Arslan, and Karaşan (2016) and it is shown that the former model cannot scale up to large networks due to 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 $\hat{\lambda} = \hat{\lambda}_q$, $q \in Q$.

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 branch-and-price algorithm in Yıldız, Arslan, and Karaşan (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, Arslan, and Karaşan (2016), the number of stations considered are 1, 5, 10, ..., 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. Two leftmost columns are parameters of the experiment: p is the number of stations to be located and 'Tol (%)' 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 (UB-OPT)/OPT$\times$100%. Note that, for B&C-2 algorithm, the upper bound we obtain at the root node is equal to the optimal value of the LP relaxation of the model, since 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.

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	1183	1183	1183	1196	1183
	10	7.7	7.3	7.9	7.2	7.3	0	0	0	0	0	0	0	0	0	0	1173	1173	1173	1207	1173
	20	14.9	11.1	11.5	10.9	11.4	0	0	0	0	0	0	0	0	0	0	1173	1173	1173	1174	1173
	50	20.3	20.6	20.4	19.9	20.8	0	0	0	9.83	0	0	0	0	16	0	1200	1200	1200	1560	1200
5	0	3.4	3.3	3.5	3601.9*	3.5	0	0	0	N/A	0	0	0	0	4765	0	2408	2408	2408	179927	2408
	10	29.9	230.2	30.8	3607.0*	31.7	0.38	0	0	N/A	0	5	0	0	2637	0	3247	4084	3189	215873	3189
	20	92.0	88.4	105.8	3615.8*	250.2	4.82	2.55	2.55	N/A	2.55	34	34	23	1931	17	3818	3818	5237	243382	4034
	50	119.3	123.4	135.9	3623.1*	330.8	0.50	0.32	0.32	N/A	0.32	9	9	7	1535	6	2712	2712	2720	391461	2753
10	0	4.1	54.9	4.0	3601.6*	4.8	0.78	0.60	0.60	N/A	0.60	5	3	3	6423	3	2362	2870	2427	122227	2226
	10	25.3	24.3	28.4	3608.1*	50.2	1.18	1.18	1.18	N/A	1.18	22	22	6	4200	6	2478	2478	2418	160256	2416
	20	49.5	3623.6*	107.4	3612.4*	494.6	0.39	N/A	0.25	N/A	0.24	25	0	40	3704	30	2630	6648	2779	230458	2467
	50	90.7	90.9	108.8	3621.3*	281.4	0.25	0.14	0.14	N/A	0.14	25	25	9	3921	8	1952	1952	2050	233232	2054
15	0	4.1	1020.7	3.7	3601.8*	4.8	0.06	0	0	N/A	0	7	0	0	5310	0	2877	3863	2865	108078	2829
	10	21.2	21.5	21.4	3607.7	20.0	0.01	0.01	0.01	N/A	0.01	4	4	4	8563	0	1938	1938	1938	102612	1938
	20	44.6	3646.7*	63.3	3612.1*	208.0	0.08	N/A	0.07	N/A	0.07	7	0	8	17552	8	2019	4005	2483	66371	1648
	50	72.2	641.0	81.5	236.9	145.8	0	0	0	0	0	0	0	0	1536	0	1392	1403	1372	50886	1372
20	0	4.5	676.4	10.7	3601.7*	43.4	0.48	0.34	0.34	N/A	0.34	72	66	30	13389	42	2556	6391	3401	66353	3332
	10	17.8	581.9	24.6	286.8	65.1	0.02	0.02	0.02	0.16	0.02	5	11	3	4466	3	1680	2136	1760	18334	1461
	20	35.1	3660.3*	97.1	1002.0	306.2	0.02	N/A	0.02	0.03	0.02	45	0	32	22452	11	1446	1869	1437	15651	1535
	50	70.8	72.2	71.3	71.5	71.7	0	0	0	0	0	0	0	0	217	0	1295	1295	1295	17770	1295
25	0	6.1	338.4	60.2	3601.7*	208.7	0.12	0.12	0.12	N/A	0.12	247	241	383	89833	223	3217	2967	2869	23058	2952
	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	1284	1284	1331	5782	1330
	20	31.4	446.1	47.4	25.9	93.0	0	0	0	0	0	5	5	5	31	0	1234	1234	1277	4096	1234
	50	72.3	71.6	71.5	38.3	74.1	0	0	0	0	0	0	0	0	10	0	1297	1297	1297	5203	1297
30	0	4.7	88.4	3.9	12.1	5.7	0	0	0	0	0	16	0	0	629	0	2456	1889	1970	7831	1907
	10	15.1	14.7	14.8	13.0	14.3	0	0	0	0	0	0	0	0	45	0	1215	1215	1215	3809	1215
	20	38.6	33.8	34.0	15.9	35.1	0	0	0	0	0	0	0	0	3	0	1283	1283	1283	1900	1283
	50	72.0	71.0	73.3	39.6	73.2	0	0	0	0	0	0	0	0	6	0	1297	1297	1297	3353	1297
35	0	2.9	2.9	3.1	3.4	2.9	0	0	0	0	0	2	2	2	11	2	1344	1344	1344	2588	1344
	10	13.9	14.9	14.9	9.2	14.7	0	0	0	0	0	0	0	0	0	0	1184	1184	1184	1562	1184
	20	34.4	33.2	33.2	21.3	33.3	0	0	0	0	0	0	0	0	0	0	1283	1283	1283	2492	1283
	50	74.2	72.7	70.1	54.7	72.2	0	0	0	0	0	0	0	0	0	0	1297	1297	1297	3700	1297
Averages		34.7	493.9	43.5	1525.0	94.3	0.28	N/A	0.16	N/A	0.16	17.0	13.5	17.7	6042.4	11.3	1873	2255	1942	71668	1853

* The algorithm terminated due to 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	1873	65.11	-
B&C-2	493.9	19.0	473.9	2255	64.82	-
B&C-3	43.5	30.9	12.2	1942	64.71	-
B&C-4	1525.0	43.6	0	71668	-	-
B&C-5	94.3	28.7	65.2	1853	64.75	17.66

The average solution times are 34.7, 493.9, 43.5, 1525.0 and 94.3 seconds for 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 with asterisk in Table 3) within one hour time limit; it terminated at the root node and the maximum optimality gap was 3.1%. Table 4 shows that 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 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 due to weak cuts being added. Therefore, it fails to solve 13 of the 32 instances within the time limit (Table 3). In B&C-1, on the other hand, the minimalization process removes, on the 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

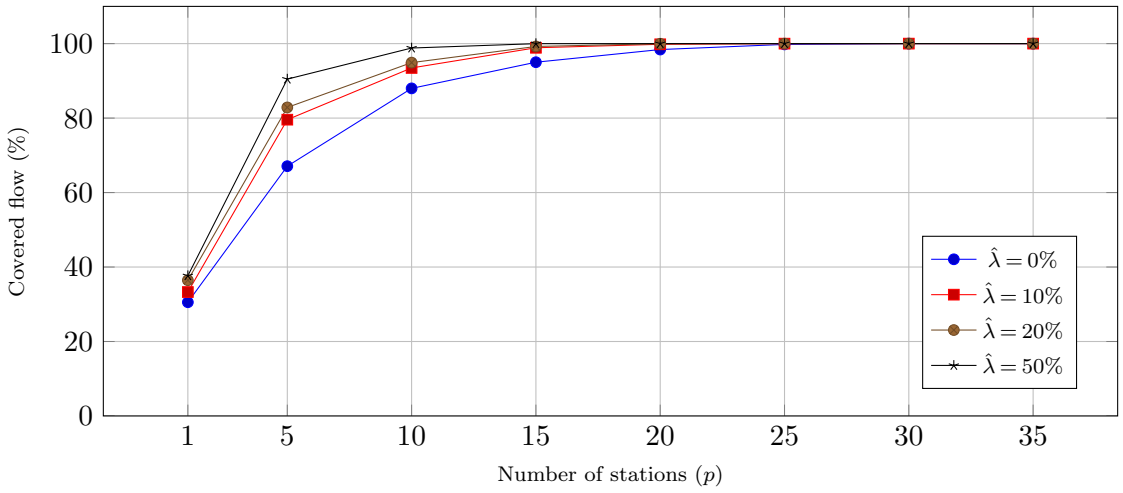
the average, minimalization removes 17.66% of the nodes from the cuts generated by the heuristic, however, this came at a cost of multiplying the fractional separation times by more than five. The average time for fractional separation increased from 12.2 seconds to 65.2 seconds. Even though the average number of cuts added is reduced from 1942 to 1853, 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 IntSep algorithm but not in 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 since 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, the B&C-1 algorithm, implementing separation only at integer solutions and 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 where 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 B&C-1 algorithm.

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, Arslan, and Karaşan (2016), the two are comparable. According to PassMark Software (2017), our computer has 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.

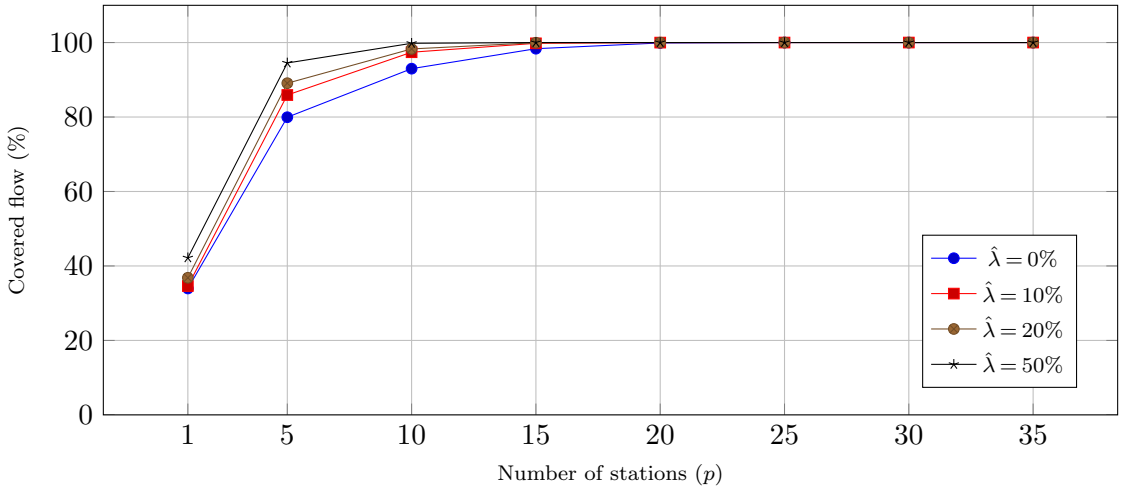
Table 5 Results for the CA instances with different vehicle ranges

		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	1183	33.95	1.5	0	0	1208	36.46	2.3	0	0	1210
	10	33.29	7.7	0	0	1173	34.62	8.2	0	0	1227	37.22	8.4	0	0	1241
	20	36.46	14.9	0	0	1173	36.83	14.0	0	0	1284	38.08	22.1	0	0	1309
	50	37.66	20.3	0	0	1200	42.19	38.6	0	0	1333	43.84	44.7	0	0	1569
5	0	67.08	3.4	0	0	2408	79.94	3.0	0	0	1849	85.18	2.7	0	0	1500
	10	79.57	29.9	0.38	5	3247	85.91	23.9	0	0	1911	90.53	17.8	0.26	5	1545
	20	82.86	92.0	4.82	34	3818	89.08	52.9	0.17	3	1927	93.87	48.5	0	0	1353
	50	90.46	119.3	0.5	9	2712	94.51	105.5	0	0	1713	97.52	100.2	0	0	1385
10	0	87.98	4.1	0.78	5	2362	92.98	3.3	0.91	28	1792	95.64	3.5	0.35	7	1604
	10	93.47	25.3	1.18	22	2478	97.4	18.6	0	0	1557	98.63	17.1	0	0	1386
	20	94.9	49.5	0.39	25	2630	98.29	46.1	0.07	7	1655	99.02	48.6	0.04	6	1355
	50	98.82	90.7	0.25	25	1952	99.8	91.7	0	0	1394	99.9	90.1	0	0	1281
15	0	95.01	4.1	0.06	7	2877	98.35	4.0	0.28	37	1872	99.22	3.5	0.25	66	1482
	10	98.89	21.2	0.01	4	1938	99.79	19.3	0.05	10	1315	99.87	16.0	0.06	11	1182
	20	99.24	44.6	0.08	7	2019	99.95	45.9	0	0	1352	99.97	47.4	0.01	2	1220
	50	100	72.2	0	0	1392	100	83.9	0	0	1188	100	101.2	0	0	1178
20	0	98.41	4.5	0.48	72	2556	99.89	4.1	0.06	92	1912	99.97	3.9	0.03	36	1424
	10	99.82	17.8	0.02	5	1680	99.98	17.5	0.01	36	1234	100	17.4	0	0	1198
	20	99.97	35.1	0.02	45	1446	100	41.3	0	0	1176	100	45.3	0	0	1177
	50	100	70.8	0	0	1295	100	82.8	0	0	1188	100	91.2	0	0	1178
25	0	99.79	6.1	0.12	247	3217	100	3.0	0	0	1356	100	2.9	0	0	1244
	10	99.99	16.4	0.01	10	1284	100	16.0	0	0	1201	100	15.8	0	0	1174
	20	100	31.4	0	5	1234	100	43.1	0	0	1171	100	42.1	0	0	1177
	50	100	72.3	0	0	1297	100	82.9	0	0	1188	100	90.2	0	0	1178
30	0	100	4.7	0	16	2456	100	3.1	0	0	1670	100	2.5	0	0	1205
	10	100	15.1	0	0	1215	100	17.5	0	0	1198	100	17.4	0	0	1169
	20	100	38.6	0	0	1283	100	40.6	0	0	1171	100	42.7	0	0	1177
	50	100	72.0	0	0	1297	100	87.7	0	0	1188	100	97.4	0	0	1178
35	0	100	2.9	0	2	1344	100	2.7	0	0	1222	100	2.7	0	0	1205
	10	100	13.9	0	0	1184	100	16.9	0	0	1198	100	15.7	0	0	1169
	20	100	34.4	0	0	1283	100	44.5	0	0	1171	100	42.5	0	0	1177
	50	100	74.2	0	0	1297	100	87.1	0	0	1188	100	90.8	0	0	1178

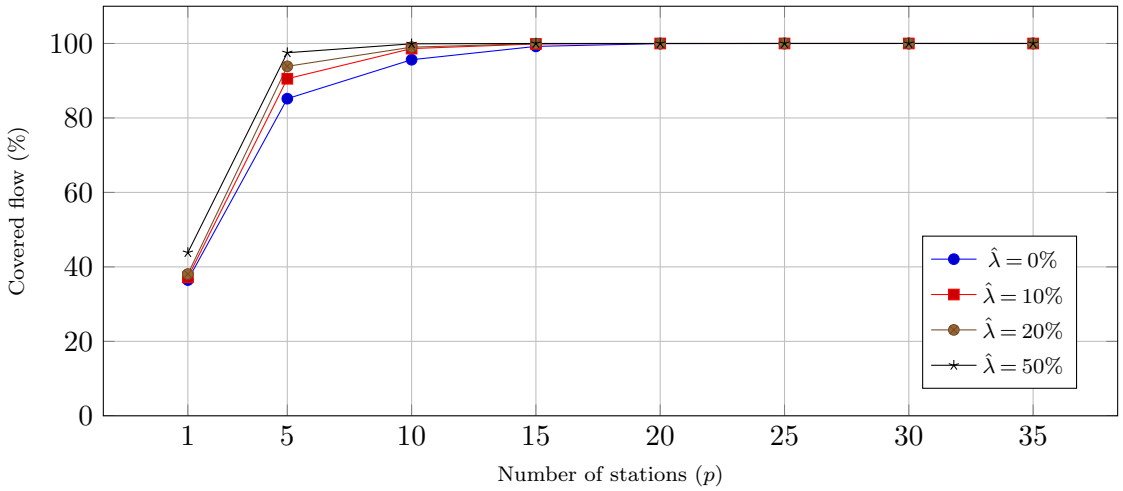
Figure 3 plots the covered vehicle flow for different deviation tolerances when $p = 1, \ldots, 35$ stations are optimally located and the vehicles have ranges of 100, 150 and 200 kilometers in parts (a), (b) and (c), respectively. Coverage increases to 100% steadily in all three plots. The rate of increase 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, on the other hand, full coverage is reached when $p = 15$. The rate of increase is also faster for higher vehicle ranges. When



(a) Range = 100 km.



(b) Range = 150 km.



(c) Range = 200 km.

Figure 3 **Covered flow as 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 California network**

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 100 kilometers 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 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% to 43.85% of the vehicle flow can be covered, depending on the range of the vehicle and deviation tolerance of the drivers. 15 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 in order 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 due to 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 due to range*.

Figure 4(a) shows the breakdown of covered and missed vehicle flows for different tolerances when 5 stations are optimally located in the CA network with vehicles of 100

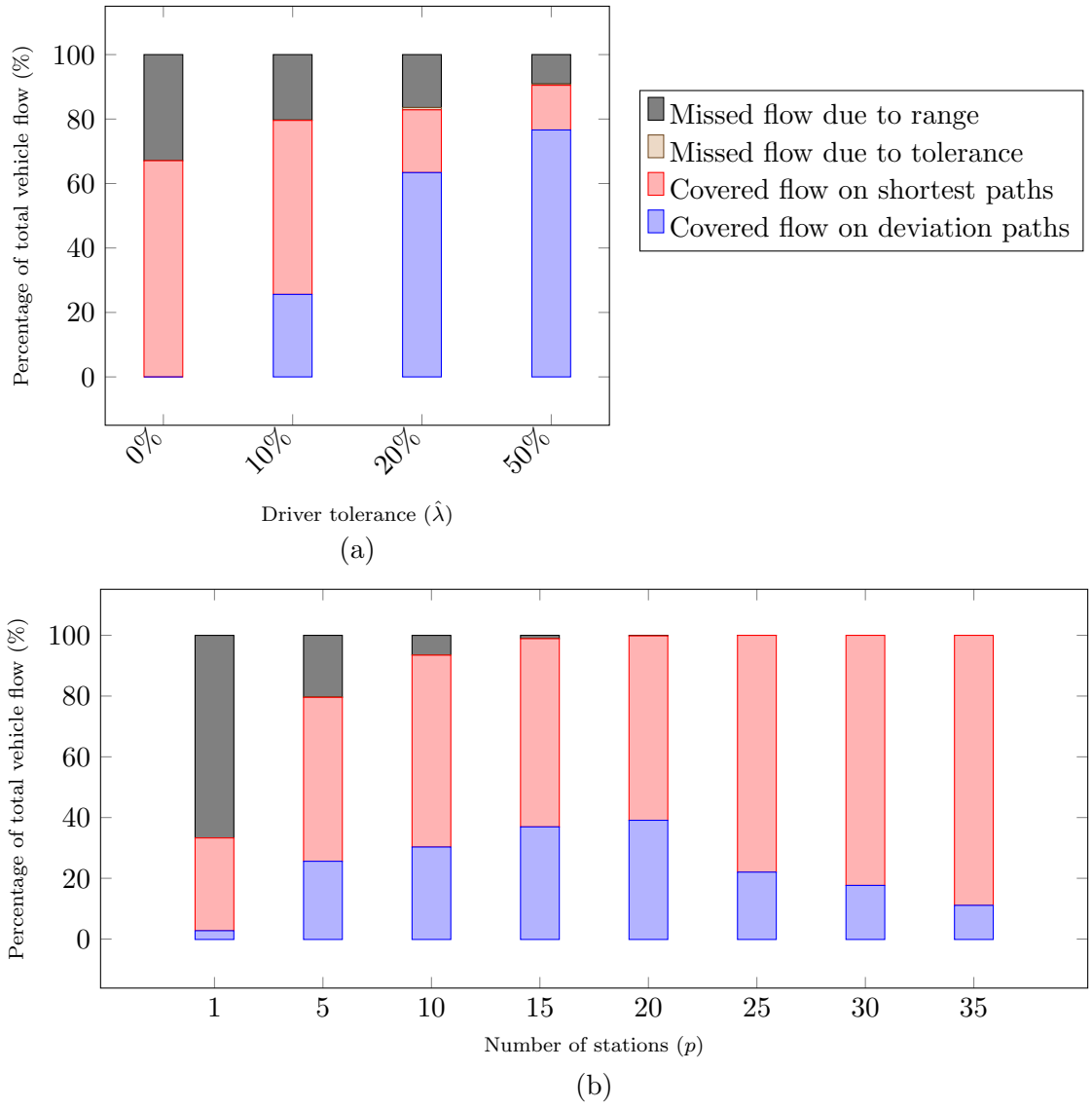


Figure 4 **(a) Percentage of covered and missed vehicle flows for different driver tolerances in the California network with vehicles of 100 km range and when $p = 5$ stations are optimally located, (b) Percentage of covered and missed vehicle flows when $p = 1, \ldots, 35$ stations are optimally located in the California network with vehicles of 100 km range and for deviation tolerance $\hat{\lambda} = 10\%$.**

kilometers 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 while 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 due to 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 CA network with vehicles of 100 kilometers range and driver deviation tolerance of 10%. The percentage of the deviating vehicle flow is increasing until a 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 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 California network for vehicles of 100 kilometers range and when $p = 5$ and $\hat{\lambda} = 50\%$. This figure is representative to show the 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 to 10 percent 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)

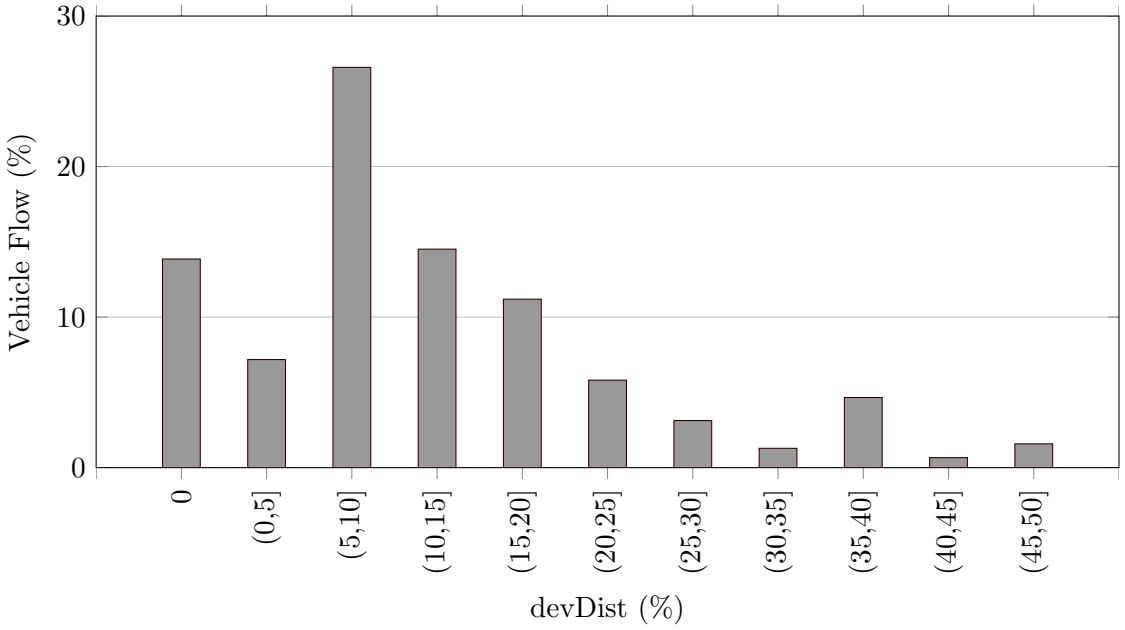


Figure 5 **Distribution of vehicle flow grouped into deviation distance intervals in California network for with vehicles of 100 kilometers range and when $p = 5$ and $\hat{\lambda} = 50\%$. The deviation distance on the horizontal axis is presented as percentage of the shortest path.**

the size of the network. In 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 CA network, there are 1167 OD pairs. The results presented in Table 5 are obtained assuming a single vehicle type travelling 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 8 vehicle types in an experiment refers to 9336 distinct demands traveling between 1167 OD pairs. The ranges of vehicles change between 100 and 275 kilometers by increments of 25. In other words, when we have 8 vehicle types running between the same OD pair, their ranges are 100, 125, ..., 275 kilometers. In this regard, Table 6 presents results for CA network for 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 1167 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 population and 30 kilometers apart, corresponding to 1874 (and 3121, respectively) OD pairs and over 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 8 minutes. Apart from these two new OD pair sets, we also test unit vehicle flows between each OD pair. In our original experiments with 1167 OD pairs, the vehicle flows are highly unbalanced, changing between 14.51 and 1,235,110 veh/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.

Table 6 Results for the CA instances with different vehicle types

		0% Driver Tolerance					10% Driver Tolerance					20% Driver Tolerance					50% Driver Tolerance				
# Veh 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	1183	33.29	7.7	0	0	1173	36.46	14.9	0	0	1173	37.66	20.3	0	0	1200
	5	67.08	3.4	0	0	2408	79.57	29.9	0	5	3247	82.86	92.0	2.55	37	4092	90.46	119.3	0.32	9	2712
	10	87.98	4.1	0.66	5	2362	93.47	25.3	1.18	22	2478	94.90	49.5	0.22	34	2696	98.82	90.7	0.15	25	1952
	15	95.01	4.1	0	7	2877	98.89	21.2	0	0	1938	99.24	44.6	0.07	7	2019	100.00	72.2	0	0	1392
	20	98.41	4.5	0.34	100	2569	99.82	17.8	0.02	5	1680	99.97	35.1	0.01	34	1444	100.00	70.8	0	0	1295
	25	99.79	6.1	0.12	274	3261	99.99	16.4	0.01	6	1284	100.00	31.4	0	5	1234	100.00	72.3	0	0	1297
	30	100.00	4.7	0	98	2340	100.00	15.1	0	0	1215	100.00	38.6	0	0	1283	100.00	72.0	0	0	1297
	35	100.00	2.9	0	2	1344	100.00	13.9	0	0	1184	100.00	34.4	0	0	1283	100.00	74.2	0	0	1297
2	1	30.54	3.2	0	0	2380	33.29	12.5	0	0	2384	36.46	22.5	0	0	2375	37.66	43.9	0	0	2518
	5	71.05	6.5	0	1	4365	81.12	48.7	0.08	2	5000	84.14	142.7	0.4	52	6150	91.61	250.2	0	14	4783
	10	88.08	6.6	1.05	15	4610	94.05	46.8	0.64	37	4515	95.54	97.5	0.25	23	4178	98.99	183.5	0	11	3432
	15	95.27	6.8	0.16	9	4683	98.90	34.1	0	7	3098	99.32	93.2	0.03	9	3289	100.00	142.7	0	0	2506
	20	98.57	10.7	0.47	104	5022	99.86	39.1	0.02	18	3275	99.97	81.9	0.02	54	3152	100.00	143.7	0	0	2524
	25	99.85	9.7	0.08	276	3399	99.99	34.4	0.01	155	2590	100.00	78.3	0	0	2636	100.00	149.8	0	0	2443
	30	100.00	8.2	0	118	3528	100.00	34.0	0	1	2694	100.00	71.8	0	0	2365	100.00	142.5	0	0	2443
	35	100.00	5.5	0	5	2963	100.00	30.0	0	0	2352	100.00	73.3	0	0	2365	100.00	150.8	0	0	2443
3	1	30.65	5.0	0	0	3592	33.41	21.7	0	0	3617	36.58	50.4	0	0	3712	38.78	70.7	0	0	3775
	5	73.76	9.5	0	0	5843	82.19	70.0	0.71	9	6383	85.00	212.4	0.94	51	8671	92.19	343.5	0.16	9	7093
	10	89.04	10.6	1.01	11	6603	94.83	74.1	0.28	25	6292	96.34	137.6	0.15	21	5357	99.07	277.7	0.09	21	4777
	15	95.77	11.5	0.36	50	6161	98.92	65.0	0	6	5398	99.40	145.1	0.02	7	5241	100.00	233.6	0	1	3779
	20	98.79	17.9	0.41	156	7380	99.87	62.7	0.02	23	4494	99.98	116.2	0.02	80	4011	100.00	237.7	0	0	3685
	25	99.89	12.2	0.06	90	5238	100.00	63.5	0	335	4076	100.00	117.6	0	0	3873	100.00	238.0	0	0	3687
	30	100.00	9.8	0	5	4655	100.00	55.9	0	1	3722	100.00	121.1	0	0	3595	100.00	237.8	0	0	3687
	35	100.00	9.2	0	1	4603	100.00	55.9	0	0	3629	100.00	119.7	0	0	3590	100.00	236.3	0	0	3687
4	1	31.56	6.1	0	0	4783	33.69	31.9	0	0	4855	36.89	51.8	0	0	5008	39.63	106.1	0	0	5110
	5	75.88	12.8	0	1	7704	83.12	94.3	0.03	9	8141	86.00	264.7	0.26	61	10653	92.90	483.8	0.1	11	9605
	10	90.11	13.9	0.64	17	7619	95.29	88.2	0.32	9	8325	96.77	204.3	0.13	3	7676	99.12	384.9	0.04	16	6029
	15	96.10	18.9	0.52	99	7945	98.97	88.7	0.06	21	7765	99.45	198.2	0.06	19	6177	100.00	327.5	0	1	4966
	20	98.95	19.1	0.38	163	7595	99.89	71.5	0.02	16	5394	99.98	175.1	0.02	152	5027	100.00	326.1	0	0	4866
	25	99.91	16.2	0.05	129	6069	100.00	72.3	0	9	4777	100.00	158.9	0	4	5235	100.00	353.3	0	0	4868
	30	100.00	16.8	0	123	6143	100.00	69.9	0	0	4882	100.00	156.3	0	0	4743	100.00	318.9	0	0	4868
	35	100.00	9.6	0	0	5089	100.00	68.2	0	0	4862	100.00	157.9	0	0	4738	100.00	332.9	0	0	4868
5	1	32.54	8.6	0	0	5993	33.91	34.9	0	0	6108	37.13	80.3	0	0	6378	40.47	146.4	0	0	6682
	5	77.53	14.6	0	0	8715	83.81	115.0	0.26	19	10447	86.83	299.8	0.22	43	11714	93.43	574.1	0.09	16	10169
	10	90.93	17.7	0.46	11	9137	95.67	113.0	0.32	7	9913	97.11	238.2	0.1	7	8246	99.24	492.9	0.04	6	7380
	15	96.56	23.0	0.47	59	9433	99.00	106.5	0.12	42	8294	99.52	222.4	0	23	7396	100.00	418.1	0	0	6165
	20	99.05	33.1	0.36	270	8884	99.90	94.1	0.02	14	6502	99.98	208.6	0.01	107	6221	100.00	488.7	0	0	6010
	25	99.93	18.0	0.05	108	7245	100.00	97.6	0	61	6285	100.00	200.6	0	0	6011	100.00	435.9	0	0	6010
	30	100.00	19.0	0	41	7701	100.00	82.7	0	0	6027	100.00	192.9	0	0	5953	100.00	463.5	0	0	6010
	35	100.00	11.2	0	0	6280	100.00	82.1	0	0	6027	100.00	186.1	0	0	5953	100.00	428.9	0	0	6010
6	1	33.44	10.0	0	0	7228	34.20	43.2	0	0	7422	37.56	106.2	0	0	7891	41.38	208.6	0	0	8339
	5	78.77	18.6	0	0	10354	84.76	149.1	0.23	19	12171	87.53	390.2	0.28	80	14545	93.81	694.3	0.1	17	12087
	10	91.56	26.3	0.36	37	11116	95.98	118.3	0.46	12	10485	97.35	275.7	0.03	8	9576	99.33	593.6	0	13	9072
	15	96.89	30.1	0.44	63	10577	99.10	127.5	0.11	47	9566	99.58	247.8	0	9	8637	100.00	538.1	0	0	7125
	20	99.14	34.3	0.34	150	10924	99.91	120.2	0.02	20	7921	99.98	260.5	0.01	130	7399	100.00	556.1	0	0	7074
	25	99.94	31.6	0.04	290	10321	100.00	108.1	0	10	7371	100.00	223.3	0	7	7184	100.00	517.0	0	0	7074
	30	100.00	20.5	0	37	8735	100.00	99.4	0	0	7123	100.00	221.9	0	0	7122	100.00	503.8	0	0	7074
	35	100.00	15.8	0	1	7498	100.00	96.5	0	0	7123	100.00	256.7	0	0	7122	100.00	491.0	0	0	7074
7	1	34.08	11.9	0	0	8498	34.73	54.7	0	0	8782	37.99	119.4	0	0	9501	42.13	287.6	0	0	10265
	5	79.91	21.6	0	0	12186	85.57	180.1	0.51	17	13729	88.17	496.3	0.26	114	17845	94.07	823.5	0.12	17	14424
	10	92.21	27.2	0.25	13	12542	96.32	148.4	0.34	14	11258	97.53	319.0	0.06	11	10846	99.39	680.2	0	24	9549
	15	97.15	48.1	0.42	152	12374	99.18	141.1	0.1	31	10539	99.63	301.8	0	0	9780	100.00	668.0	0	0	8292
	20	99.23	50.6	0.31	418	10563	99.92	145.4	0.02	22	9065	99.99	299.9	0.01	122	8594	100.00	639.2	0	0	8241
	25	99.94	34.4	0.03	257	9954	100.00	122.3	0	27	8478	100.00	315.9	0	0	8320	100.00	644.0	0	0	8241
	30	100.00	23.2	0	22	9180	100.00	125.7	0	0	8290	100.00	276.0	0	0	8289	100.00	623.5	0	0	8241
	35	100.00	17.6	0	3	8671	100.00	112.4	0	0	8290	100.00	278.1	0	0	8289	100.00	608.6	0	0	8241
8	1	34.88	15.3	0	0	9766	35.56	63.5	0	0	10092	38.60	157.1	0	0	11120	43.01	367.1	0	0	12051
	5	80.84	25.8	0	1	13565	86.18	174.5	0.41	19	14778	88.65	479.9	0.96	88	17542	94.47	1007.2	0.09	21	16476
	10	92.71	29.3	0.23	14	13144	96.60	162.0	0.18	13	12504	97.71	390.3	0.21	16	11967	99.44	765.5	0	7	10829
	15	97.35	51.0	0.43	162	13216	99.25	166.6	0.1	26	11685	99.66	355.1	0	0	10923	100.00	681.7	0	0	9459
	20	99.30	41.7	0.29	204	12144	99.93	144.6	0.02	7	9970	99.99	363.7	0.01	115	9856	100.00	655.3	0	0	9408
	25	99.95	33.4	0.03	184	10896	100.00	149.3	0	8	9843	100.00	361.8	0	0	9487	100.00	672.2	0	0	9408
	30	100.00	38.5	0	295	10147	100.00	137.7	0	0	9457	100.00	338.0	0	0	9456	100.00	714.4	0	0	9408
	35	100.00	21.7	0	1	9830	100.00	134.5	0	0	9457	100.00	333.9	0	0	9456	100.00	633.1	0	0	18457

Table 7 Results for the CA instances with different OD pair numbers

		# ODpairs = 1167 (Total Flow = 5,718,328 veh/year)					# ODpairs = 1874 (Total Flow = 5,912,295 veh/year)					# ODpairs = 3121 (Total Flow = 6,115,458 veh/year)					# ODpairs = 1167 (Unit Flow for each OD pair)				
p	Tol (%)	Opt value (1000 veh/year)	Sol time (s)	Root node gap (%)	nNodes	nCuts	Opt value (1000 veh/year)	Sol time (s)	Root node gap (%)	nNodes	nCuts	Opt value (1000 veh/year)	Sol time (s)	Root node gap (%)	nNodes	nCuts	Opt value	Sol time (s)	Root node gap (%)	nNodes	nCuts
1	1	1747	1.9	0	0	1183	1748	2.7	0	0	1893	1770	4.2	0	0	3138	77	1.8	0	0	1169
	1.1	1903	7.7	0	0	1173	1928	10.9	0	0	1885	1972	19.6	0	0	3138	89	7.3	0	0	1170
	1.2	2085	14.9	0	0	1173	2110	23.7	0	0	1885	2145	29.4	0	0	3139	98	12.1	0	0	1170
	1.5	2154	20.3	0	0	1200	2183	44.5	0	0	1918	2226	55.1	0	0	3196	114	20.1	0	0	1181
5	1	3836	3.4	0	0	2408	3877	4.9	0	1	3707	3946	8.8	0	0	6041	387	3.7	16.54	21	2834
	1.1	4550	29.9	0.38	5	3247	4620	53.8	0.7	5	5223	4719	98.2	1.42	7	8857	524	32.2	8.68	116	3367
	1.2	4738	92.0	4.82	34	3818	4812	125.6	0.95	25	5522	4919	234.4	1.60	43	9771	625	83.1	7.73	132	3839
	1.5	5173	119.3	0.50	9	2712	5279	285.7	0.71	33	6586	5381	467.1	1.13	77	9902	794	247.8	6.52	125	6431
10	1	5031	4.1	0.78	5	2362	5145	5.4	0.59	5	3489	5267	9.0	0.63	5	5991	699	4.0	1.96	21	3015
	1.1	5345	25.3	1.18	22	2478	5472	41.7	0.43	11	3643	5617	91.8	0.38	24	7200	913	24.0	2.37	28	2349
	1.2	5427	49.5	0.39	25	2630	5576	93.5	0.16	56	4092	5715	166.3	0.69	61	7152	1022	38.3	0.44	25	1937
	1.5	5651	90.7	0.25	25	1952	5811	180.0	0.28	41	3518	5962	351.9	0.29	13	6681	1132	88.1	0.39	5	1936
15	1	5433	4.1	0.06	7	2877	5574	5.9	0.1	0	4263	5707	11.2	0	1	7002	939	4.2	2.24	7	3000
	1.1	5655	21.2	0.01	4	1938	5800	34.2	0	1	2829	5952	72.4	0	3	5347	1116	18.6	0.09	0	1600
	1.2	5675	44.6	0.08	7	2019	5833	92.9	0.17	21	3590	6005	126.9	0.07	12	5051	1146	33.6	0.09	5	1396
	1.5	5718	72.2	0	0	1392	5903	208.1	0.05	19	3751	6092	257.1	0.05	2	5098	1167	70.8	0	0	1348
20	1	5627	4.5	0.48	72	2556	5775	7.7	0.42	71	4647	5915	11.0	0.47	27	6684	1101	4.8	1.35	50	3240
	1.1	5708	17.8	0.02	5	1680	5871	34.0	0.03	12	2841	6041	62.2	0.02	3	4745	1154	16.9	0.11	6	1396
	1.2	5717	35.1	0.02	45	1446	5897	80.8	0.07	32	3120	6080	133.9	0.03	2	4643	1162	33.7	0.17	14	1421
	1.5	5718	70.8	0	0	1295	5912	122.0	0	1	2300	6114	203.7	0	1	3576	1167	73.9	0	0	1295
25	1	5707	6.1	0.12	247	3217	5875	8.7	0.19	227	4072	6025	13.3	0.11	98	6489	1147	4.7	0.93	301	2444
	1.1	5718	16.4	0.01	10	1284	5902	29.9	0.06	71	2371	6086	57.2	0.03	13	4272	1166	15.3	0.09	10	1248
	1.2	5718	31.4	0	5	1234	5910	73.3	0.01	54	2576	6108	113.5	0.02	30	4092	1167	32.4	0	5	1234
	1.5	5718	72.3	0	0	1297	5912	117.1	0	0	2149	6115	210.7	0	0	3475	1167	71.3	0	0	1297
30	1	5718	4.7	0	16	2456	5902	7.7	0.09	495	3473	6072	12.3	0.15	115	6235	1167	3.6	0	0	1897
	1.1	5718	15.1	0	0	1215	5912	29.6	0	20	2317	6108	52.7	0.01	6	4160	1167	15.5	0	0	1215
	1.2	5718	38.6	0	0	1283	5912	75.0	0	41	2504	6114	105.8	0	29	3773	1167	34.3	0	0	1283
	1.5	5718	72.0	0	0	1297	5912	119.8	0	0	2138	6115	201.5	0	0	3373	1167	72.6	0	0	1297
35	1	5718	2.9	0	2	1344	5911	5.3	0	63	2827	6102	12.4	0.13	226	5720	1167	3.1	0	2	1344
	1.1	5718	13.9	0	0	1184	5912	27.6	0	0	2061	6114	53.7	0	12	3990	1167	15.7	0	0	1184
	1.2	5718	34.4	0	0	1283	5912	63.5	0	3	2221	6115	107.6	0	14	3746	1167	32.7	0	0	1283
	1.5	5718	74.2	0	0	1297	5912	118.2	0	0	2138	6115	217.1	0	0	3288	1167	72.4	0	0	1297

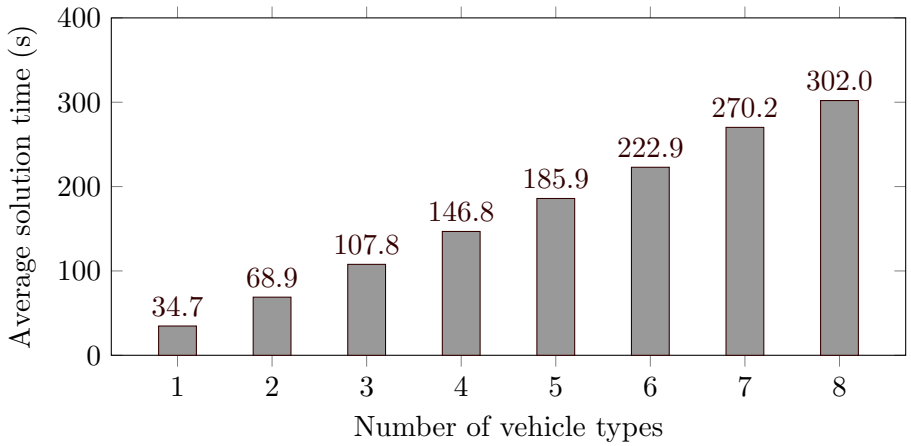


Figure 6 **Solution times for different number of vehicle types**

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, 1000, 2000 and 4000 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. '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 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 1000 nodes and 4000 OD pairs.

Table 8 Results for randomly generated graphs

			0% Driver Tolerance						10% Driver Tolerance						20% Driver Tolerance					
Network	# 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	1000	10	14.91	0	2.0	11.78	164	1781	23.15	0	1.7	6.63	46	1608	27.35	0	3.5	6.68	134	1682
		20	29.69	0	3.3	7.58	510	1752	43.56	0	1.9	1.57	41	1643	50.54	0	3.5	0.52	19	1577
		30	43.51	0	2.7	4.47	298	1728	60.82	0	1.8	0.12	4	1610	67.4	0	3.4	0.59	38	1590
		50	67.41	0	1.8	0.47	69	1743	85.96	0	2.1	0.25	14	1467	91.87	0	3.1	0.06	4	1324
		100	94.71	0	1.3	0.00	1	1320	100	0	1.7	0.00	0	1152	100	0	3.1	0.00	0	1075
	2000	10	15.06	0	4.4	10.09	254	3315	23.69	0	3.5	4.49	25	2982	27.74	0	5.4	4.71	75	2886
		20	29.54	0	6.8	6.40	435	3360	44.89	0	3.4	1.10	23	3098	50.11	0	5.6	0.68	17	2890
		30	42.75	0	10.1	5.19	848	3449	61.63	0	3.6	0.56	24	3153	68.02	0	6.1	0.02	3	3071
		50	65.82	0	3.7	0.99	30	3229	87.18	0	3.5	0.10	6	2891	92.09	0	6.3	0.14	5	2665
		100	91.91	0	3.5	0.24	58	2615	100	0	2.8	0.00	0	2113	100	0	4.0	0.00	0	2097
	4000	10	14.72	0	15.0	12.33	372	6604	24.37	0	7.2	2.92	12	6101	29.02	0	10.2	2.23	27	5614
		20	29.86	0	15.9	4.94	415	6569	45.48	0	8.1	0.78	25	6071	51.6	0	11.9	0.72	17	6004
		30	43.92	0	9.5	1.24	40	6622	61.95	0	8.1	0.05	3	6103	68.12	0	11.7	0.40	10	5822
		50	64.58	0	14.2	1.88	227	6565	86.6	0	7.0	0.00	1	5236	91.34	0	11.4	0.00	1	4997
		100	91.43	0	9.3	0.41	80	5436	100	0	6.3	0.00	0	4231	100	0	8.9	0.00	0	4224
G-500	1000	10	10.91	0	3.4	9.64	184	1676	16.27	0	5.7	8.56	618	1609	18	0	14.1	11.70	2884	1616
		20	20.99	0	3.2	4.68	238	1593	30.33	0	4.7	3.26	171	1559	33.58	0	10.1	2.69	247	1586
		30	29.38	0	4.3	3.77	520	1634	42.13	0	5.2	1.94	132	1590	46.73	0	10.2	0.74	37	1570
		50	44.8	0	5.7	3.40	869	1733	61.27	0	5.1	0.84	28	1619	66.17	0	10.9	0.96	210	1570
		100	73.59	0	4.4	0.96	165	1769	93.02	0	4.9	0.11	8	1346	96.47	0	11.6	0.00	1	1273
	2000	10	9.67	0	9.7	16.99	596	3204	16.01	0	10.5	10.59	295	3061	18.34	0	19.9	8.57	470	3164
		20	19.03	0	8.2	7.24	373	3162	30.23	0	9.1	1.67	55	3006	34.47	0	17.7	2.19	87	3048
		30	27.03	0	15.2	4.22	885	3241	41.8	0	10.5	1.74	159	3017	47.33	0	20.9	1.92	159	3053
		50	41.13	0	51.9	3.88	5730	3426	60.83	0	11.8	1.18	291	2949	67.48	0	18.1	0.32	43	2857
		100	69.26	0	48.3	2.22	4124	3393	90.72	0	10.7	0.11	18	2662	94.72	0	18.9	0.04	17	2477
	4000	10	10.38	0	11.0	9.55	104	6181	17.13	0	13.9	2.18	34	5706	19.74	0	35.3	2.06	33	5849
		20	18.91	0	40.5	6.77	1014	6528	29.99	0	20.1	1.89	86	6314	34.11	0	44.0	2.51	387	5930
		30	25.82	0	252.4	6.81	9115	6755	40.45	0	22.6	2.39	415	6026	45.37	0	73.9	2.93	1767	6025
		50	38.94	0	3604.6*	N/A	116059	6936	58.69	0	40.0	1.65	963	6188	64.22	0	63.7	1.53	1323	5794
		100	67.19	0	160.5	2.18	5628	6601	89.6	0	17.2	0.04	5	5200	93.67	0	36.8	0.06	21	4737
G-750	1000	10	7	0	7.7	19.28	511	1844	10.85	0	10.7	14.98	485	1934	13.36	0	24.9	12.92	1588	1829
		20	13.99	0	8.7	10.35	589	1854	21.34	0	11.7	7.62	402	1845	24.93	0	32.5	8.74	1967	1951
		30	20.65	0	8.2	6.04	476	1886	30.13	0	15.8	5.94	1710	1832	35.06	0	47.4	6.20	4387	2052
		50	32.14	0	31.4	6.11	5978	1951	45.89	0	25.1	4.04	3450	1994	53.07	0	27.7	2.49	1024	1949
		100	57.27	0	18.6	2.14	2544	1978	78.31	0	10.1	0.45	63	1771	84.85	0	23.3	0.58	300	1619
	2000	10	6.64	0	13.4	18.66	544	3468	10.98	0	22.5	14.56	790	3559	13.8	0	48.4	13.82	1394	3421
		20	12.89	0	26.1	12.41	1448	3616	20.51	0	31.1	6.98	1745	3612	25.46	0	69.5	5.80	3650	3590
		30	18.6	0	47.2	8.34	3466	3592	28.59	0	164.7	7.00	19547	3748	35.06	0	127.5	4.62	10518	3623
		50	29.41	0	82.8	5.80	6322	3743	44.23	0	47.1	3.31	3606	3473	51.76	0	62.1	2.12	2432	3371
		100	52.56	0	180.6	2.93	15147	3719	73.41	0	20.7	0.87	527	3312	81.52	0	40.2	0.25	26	3082
	4000	10	5.9	0	93.1	23.48	1750	7316	10.34	0	89.9	16.24	3599	6815	12.81	0	152.4	16.59	2785	7209
		20	11.41	0	923.9	14.08	25600	7116	20.56	0	42.0	3.94	163	7016	24.93	0	103.2	4.59	752	6862
		30	16.59	0.03	3609.3*	N/A	80559	7295	28.34	0	171.0	5.47	3599	7292	34.31	0	113.9	2.63	507	7024
		50	26.79	0.03	3609.0*	N/A	65399	7274	42.6	0	1764.8	4.62	53543	7162	49.84	0	1073.1	2.88	37632	7084
		100	50.04	0.01	3609.7*	N/A	75809	7392	73.32	0	45.4	0.49	298	6318	80.21	0	84.4	0.16	50	5818
G-1000	1000	10	4.95	0	13.3	30.32	748	1958	7.51	0	52.9	45.00	5498	2392	9.03	0	112.1	36.04	12696	2465
		20	9.82	0	21.5	14.79	2123	1966	15.53	0	39.5	14.74	4002	2225	18.46	0	229.3	18.90	30193	2376
		30	14.71	0	30.9	10.15	3244	2125	23.93	0	23.3	9.68	1552	2119	28.37	0	85.7	9.01	6824	2284
		50	24.08	0	47.5	6.55	6003	2030	38.64	0	36.9	3.64	3251	2103	44.83	0	208.9	4.66	20513	2303
		100	44.36	0	655.7	4.56	113827	2089	67.18	0	27.2	1.15	2634	2073	74.99	0	186.2	1.67	33113	2129
	2000	10	4.58	0	24.9	28.94	610	3696	6.91	0	117.4	45.38	6214	4175	8.02	0	843.9	50.80	36722	4884
		20	8.74	0	164.3	20.31	8980	3860	14.95	0	111.1	15.93	4774	4055	17.96	0	443.2	16.06	22291	4313
		30	13.22	0	272.6	13.82	14575	4076	22.14	0	464.6	10.55	28069	3903	27.3	0	245.7	7.02	10997	4087
		50	21.9	0	1858.6	8.74	123478	3756	35.16	0.01	3612.3*	N/A	169123	4071	41.93	0	2862.0	6.11	171187	4297
		100	40.84	0.02	3610.4*	N/A	145421	4078	62.1	0	306.6	1.93	19311	3958	69.67	0	321.8	1.67	21154	3760
	4000	10	3.63	0	1120.0	46.67	14060	8392	5.73	0	3184.2	49.99	65238	8680	6.71	0.26	3632.2*	N/A	59069	8489
		20	7.58	0	3349.8	26.70	58719	7949	12.62	0.05	3622.7*	N/A	59302	8388	15.66	0.10	3645.2*	N/A	57580	8088
		30	11.16	0.08	3616.7*	N/A	49983	7835	19.95	0.02	3623.7*	N/A	80167	7779	24.63	0.05	3645.3*	N/A	60854	7778
		50	18.28	0.11	3616.3*	N/A	47151	7853	33.42	0.01	3622.5*	N/A	86946	7570	40.51	0.01	3645.2*	N/A	66932	7730
		100	37.04	0.06	3616.1*	N/A	48154	7807	58.96	0.01	3622.9*	N/A	87638	7256	67.97	0	165.8	1.00	1828	7278

* Instances terminated due to time limit.

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 branch-and-cut 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 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, Arslan, and Karaşan (2016). We can address practically sized problem instances that could only be solved previously by a branch-and-price algorithm in a three-hour time frame, under two minutes. Our approach can also address multiple vehicle types and possible non-simple 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

This work was partially supported by the Scientific and Technological Research Council of Turkey (TUBITAK) under the grant number 214M211. The third author's research was partially supported by the grant of TUBITAK, programme 2221. The authors thank the associate editor and two referees for their constructive comments that helped improve the quality of this paper.

Appendix. Proofs of Propositions in Section 3 Polyhedral Analysis

In this appendix, we prove Propositions 1 through 6 presented in Section 2.2. Recall that $\mathcal{X}$ is the feasible set of the natural formulation given by (2)-(5). Let e_i the i^{th} unit vector of size $|N|$ and h_q the q^{th} unit vector of size $|Q|$. In the following, we refer to a solution in $\mathcal{X}$ as (x, y).

Proof of Proposition 1 Consider the following solutions of $\mathcal{X}$:

$$\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} e_i \\ 0 \end{pmatrix} i \in N, \begin{pmatrix} x^q \\ h_q \end{pmatrix} q \in Q,$$

where $x^q = \sum_{i \in N(\pi)} e_i$ and π is a path in $\mathcal{P}'_q$. These constitute a set of $|N| + |Q| + 1$ solutions of the RSLP-R, which are affinely independent. Hence, the dimension of $\text{conv}(\mathcal{X})$ is $|N| + |Q|$. □

Proof of Proposition 2 Consider the $|N| + |Q|$ solutions given in the proof of Proposition 1 that satisfy $y_q = 0$. Obviously, these solutions are affinely independent and they lie on the face of $\text{conv}(\mathcal{X})$ defined by inequality $y_q \geq 0$. □

Proof of Proposition 3 Suppose that the condition is satisfied. Consider the following $|N| + |Q|$ solutions of $\mathcal{X}$:

$$\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} e_j \\ 0 \end{pmatrix} j \in N \setminus \{i\}, \begin{pmatrix} x^q \\ h_q \end{pmatrix} q \in Q,$$

where $x^q = \sum_{j \in N(\pi)} e_j$ and π is a path in $\mathcal{P}'_q$ with $i \notin N(\pi)$ (such a path exists since $\{i\}$ is not a q-node-cut for $\mathcal{P}'_q$). Observe that these solutions are in $\mathcal{X}$, satisfy $x_i = 0$ and are affinely independent.

If there exists a $q \in Q$ such that $\{i\}$ is a q-node-cut for $\mathcal{P}'_q$, then $x_i \geq y_q$ is valid for $\mathcal{X}$ and dominates $x_i \geq 0$. Hence the latter cannot define a facet. □

Proof of Proposition 4 Assume that the conditions are satisfied and consider the following $|N| + |Q|$ solutions:

$$\begin{pmatrix} e_i \\ 0 \end{pmatrix}, \begin{pmatrix} e_i + e_j \\ 0 \end{pmatrix} j \in N \setminus \{i\}, \begin{pmatrix} x^q \\ h_q \end{pmatrix} q \in Q,$$

where $x^q = \sum_{j \in N(\pi^q) \cup \{i\}} e_j$ and π^q is a path in $\mathcal{P}'_q$ such that $|N(\pi^q) \cup \{i\}| \leq p$. So, these vectors are solutions of the RSLP-R, satisfy $x_i = 1$ and are affinely independent.

If $p = 1$, then $x(N) \leq 1$ dominates $x_i \leq 1$. If there exists $q \in Q$ such that $|N(\pi^q) \cup \{i\}| > p$ for all $\pi^q \in \mathcal{P}'_q$, then $x_i + y_q \leq 1$ is a valid inequality and it dominates $x_i \leq 1$. □

Proof of Proposition 5 It suffices to exhibit $|N| + |Q|$ affinely independent solutions of $\mathcal{X}$ satisfying $x(N) \leq p$ with equality. Suppose that $p < |N|$. For each $q \in Q$, we can find a solution

$$\begin{pmatrix} x^q \\ h_q \end{pmatrix}$$

where x^q is such that $x^q(N) = p$, $x^q(S) \geq 1$ for all $S \in \Gamma_q$. Consider additional $|N|$ solutions:

$$\begin{pmatrix} w_{p+1} - e_i \\ 0 \end{pmatrix} i = 1, \ldots, p-1$$

and

$$\begin{pmatrix} w_{p-1}+e_i \\ 0 \end{pmatrix} i=p,\ldots,n,$$

where $w_j=\sum_{l=1}^{j} e_l$. Observe that all these $|N|+|Q|$ solutions are in $conv(\mathcal{X})$, satisfy $x(N)=p$ and are affinely independent. Thus the face of $conv(\mathcal{X})$ defined by inequality $x(N)\leq p$ has dimension $|N|+|Q|-1$ and is a facet.

If $p=|N|$, then all solutions in $conv(\mathcal{X})$ that satisfy $x(N)=p$ also satisfy $x_i=1$ for $i\in N$. As $conv(\mathcal{X})$ is full dimensional and $x(N)\leq p$ is not a positive multiple of $x_i\leq 1$, $x(N)\leq p$ cannot be facet defining. □

Proof of Proposition 6 Suppose that the conditions are satisfied. Consider the following $|N|+1$ solutions:

$$\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} e_i \\ 0 \end{pmatrix} i\in N\setminus S, \begin{pmatrix} x^i \\ h_q \end{pmatrix} i\in S,$$

where $x^i=\sum_{j\in N(\pi_i)} e_j$ and π_i is a path in $\mathcal{P}'_q$ with $N(\pi_i)\cap S=\{i\}$. In other words, we construct a path for demand q that contains only node i from set S. Note that S being a minimal q-node-cut for $\mathcal{P}'_q$ ensures that we have such a path $\pi_i\in\mathcal{P}'_q$. Next, we describe additional $|Q|-1$ solutions for every $\hat{q}\in Q\setminus\{q\}$ depending on the two cases. If there exists a path $\pi^{\hat{q}}\in\mathcal{P}'_{\hat{q}}$ with $N(\pi^{\hat{q}})\cap S=\emptyset$, then we construct the solution

$$\begin{pmatrix} x^{\hat{q}} \\ h_{\hat{q}} \end{pmatrix}$$

where $x^{\hat{q}}=\sum_{j\in N(\pi^{\hat{q}})} e_j$. Note that $x^{\hat{q}}(S)=0$ in this case. If such a path does not exist, then by assumption, there exists 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 $|N(\pi^q)\cup N(\pi^{\hat{q}})|\leq p$. In this case we construct the solution

$$\begin{pmatrix} x^{\hat{q}} \\ h_q+h_{\hat{q}} \end{pmatrix}$$

where $x^{\hat{q}}=\sum_{j\in N(\pi^q)\cup N(\pi^{\hat{q}})} e_j$. Observe that these $|Q|-1$ solutions together with the previous $|N|+1$ solutions are affinely independent, are in $conv(\mathcal{X})$ and satisfy $y_q=x(S)$. Hence, the inequality $y_q\leq x(S)$ is facet defining for $conv(\mathcal{X})$.

If the conditions are not satisfied, then $y_q+y_{\hat{q}}\leq x(S)$ is a valid inequality for $conv(\mathcal{X})$. As this new inequality dominates $y_q\leq x(S)$, the latter cannot be facet defining for $conv(\mathcal{X})$. □

References

Arslan O, Karaşan OE, 2016 *A Benders decomposition approach for the charging station location problem with plug-in hybrid electric vehicles. Transportation Research Part B: Methodological* 93, Part A: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(0):168–178.

Baier G, Erlebach T, Hall A, Köhler E, Schilling H, Skutella M, 2006 *Length-bounded cuts and flows. International Colloquium on Automata, Languages, and Programming*, 679–690 (Springer).

Capar I, Kuby M, Leon VJ, Tsai YJ, 2013 *An arc-cover path-cover formulation and strategic analysis of alternative-fuel station locations. European Journal of Operational Research* 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 in regional science* 32(1):101–118.

Energy Information Administration, 2017a *Alternative fuel vehicle data*. URL `http://www.eia.gov/renewable/afv/users.php?fs=a`, accessed Jan 17, 2017.

Energy Information Administration, 2017b *Annual energy outlook*. URL `http://www.eia.gov/outlooks/aeo/`, accessed May 18, 2014.

European Comission, 2013 *Clean power for transport*. URL `http://europa.eu/rapid/press-release\_MEMO-13-24\_en.htm`, accessed Jan 17, 2017.

European Union, 2009 *Renewable energy directive*. URL `eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32009L0028`, accessed Jan 16, 2017.

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

IBM, 2014 *ILOG CPLEX Optimization Studio*. URL `cplex.com`, accessed Feb 13, 2017.

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

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

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

Li X, Aneja Y, 2017 *Regenerator location problem: Polyhedral study and effective branch-and-cut algorithms. European Journal of Operational Research* 257(1):25–40.

Lim S, Kuby M, 2010 *Heuristic algorithms for siting alternative-fuel stations using the flow-refueling location model. European Journal of Operational Research* 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. Mathematical Programming* 124(1-2):271–284.

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

Naveh B, et al., 2008 *JGraphT*. URL `jgrapht.org`, accessed Feb 13, 2017.

PassMark Software, 2017 *CPU performance comparison*. URL `https://www.cpubenchmark.net/compare.php?cmp[]=2052&cmp[]=1846`, accessed May 18, 2017.

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

Wang YW, Lin CC, 2009 *Locating road-vehicle refueling stations. Transportation Research Part E: Logistics and Transportation Review: Logistics and Transportation Review* 45(5):821–829.

Wang YW, Lin CC, 2013 *Locating multiple types of recharging stations for battery-powered electric vehicle transport. Transportation Research Part E: Logistics and Transportation Review: Logistics and Transportation Review* 58(0):76–87.

Wang YW, Wang CR, 2010 *Locating passenger vehicle refueling stations. Transportation Research Part E: Logistics and Transportation Review: Logistics and Transportation Review* 46(5):791–801.

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

Yıldız B, Arslan O, Karaşan OE, 2016 *A branch and price approach for routing and refueling station location model. European Journal of Operational Research* 248(3):815–826.

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