A routing model and solution approach for alternative fuel vehicles with consideration of the fixed fueling time

Yihuan Shao (yihuansh@usc.edu), Maged Dessouky (maged@usc.edu)
Department of Industrial and Systems Engineering, University of Southern California

Abstract

In this paper, we introduce the Vehicle Routing Problem for Alternative Fuel Vehicles with Fixed Fueling Time (VRPAFVFFT) to model decisions to be made with regards to the vehicle routes considering fuel stations. A Mixed Integer Programming (MIP) model is proposed with improving methods, which includes an adaptive branch-and-cut algorithm. To solve large scale problems, we present a hybrid heuristic method, which combines an Adaptive Large Neighborhood Search (ALNS) with a local search and a MIP model. We conduct experiments to show the efficiency of our solving methods as well as insights from real world cases.

Keywords: Green VRP, Alternative Fuel Vehicles, Fuel Stations, Branch-and-cut Algorithm, ALNS

1 Introduction

Transportation plays an important role in a modern society. It is essential for economic development. Globalization and materialization of the economy have resulted in an increasing demand in transportation. However transportation also brings in some problems such as noise and congestion, with resulting negative impact on the environment. In 2013, among those major energy end-use sectors (residential, commercial and industrial) in the United States, transportation accounted for a large part of the CO₂, CH₄ and N₂O emissions from fossil fuel combustions. Among various transportation sources, light duty vehicles were a main factor. Light duty vehicles increased 35% in the number of vehicle miles traveled from 1990 to 2013 and represented 60% of CO₂ emissions from fossil fuel combustions in 2013 [EPA, 2011]. Globally transportation accounted for 22% of direct and 1% of indirect (due to electricity consumption) CO₂ emissions in 2011 [Nejad et al., 2015]. All these factors call for a better planning on transportation.

Transportation has many facets at multiple decision making levels. One of the efforts being made is to exploit greener alternative fuel energies such as biodiesel, electricity, ethanol, hydrogen, methanol, natural gas and propane [Erdogan and Miller-Hooks, 2012]. Within these alternative fuel energies, Compressed Natural Gas (CNG) is considered as one of the possible solutions for fossil fuel substitution because of its
wide availability, engine compatibility, and low operations cost (Khan et al., 2015). More importantly, life cycle analysis/assessment (LCA) of CNG vehicles shows they can reduce Greenhouse Gas (GHG) emissions (Shahraeeni et al., 2015; Tong et al., 2015). In the following sections, we specifically take CNG trucks as one representation of the AFVs.

The first natural gas powered vehicle can be dated back to the 1930s. Since then renewed attention is being given to CNG vehicles especially when gasoline prices rise (Khan et al., 2015). Today there are millions of CNG vehicles around the world. With the increased emphasis on sustainability, some government systems and logistics companies have already purchased CNG vehicles to support their daily operations (Dessouky and Wang, 2009). In 2015, CNG vehicles accounted for 14% of the total Alternative Fuel Vehicles (AFV) inventory in the U.S. and contributed 55% of the petroleum savings due to AFV (Johnson and Singer, 2016).

Though CNG vehicles and other AFVs have many advantages as compared to the fossil fuel vehicles, they still face several challenges on the way to replacing traditional diesel or gasoline fuel. One of the biggest challenges is the sparse, uneven allocation of CNG fuel stations, which may lead to a long detour for refueling. In fact, the detour to a CNG fuel station could increase overall energy consumption and finally offset the low emission benefit from the CNG fuel. The long waiting time in CNG fuel stations is also another important problem. Additional challenges exist like limited tank capacity (Erdoğan and Miller-Hooks, 2012; Salimifard and Raeesi, 2014). For instance, the Freightliner M2 tractor is estimated to have a range of 275 to 325 miles per fill (Laughlin and Burnham, 2016) and it is hard to extend further because of the large gas tank volume.

Some big institutions build their private CNG fuel stations. However, many of the local freight delivery operations are performed by independent truckers who are too small to manage their own fuel stations and the public fuel stations are the only option for them. In Southern California, even for drivers who have CNG fuel stations in the base depots, they are also reliant on public stations for half of their refueling needs (Kelley and Kuby, 2015). Empirical research shows that CNG vehicle drivers do consider station locations as the most important reason for choosing a fuel station (Kelley and Kuby, 2015). Similar problems can be found for other AFVs like electric vehicles. Thus, it becomes imperative to develop efficient deployment strategies for refueling these trucks that will have minimal impact on their routing operations. We introduce the Vehicle Routing Problem for Alternative Fuel Vehicles with Fixed Fueling Time (VRPAFVFFT) to model decisions to be made with regards to the vehicle routes including the choice of fuel stations. The VRPAFVFFT is a special case of the well-known Vehicle Routing Problem (VRP). It combines classical restrictions such as capacity as well as new constraints for AFVs, such as fuel tank capacity, waiting time for refueling and refueling time. In our model, we specifically introduce the fixed fueling time, which mainly comes from the
long waiting time at the fuel stations, especially for CNG fuel stations (Ahmed et al., 2013; Khan et al., 2015; U.S. Department of Energy, 2017b). It occurs every time a truck visits a fuel station and is independent of the gas volume a truck refuels. Since the fixed fueling time can be significant, every time a vehicle visits a fuel station, the time available to perform its routing is reduced, so the number of times an AFV visits a fuel station should also be minimized. This fixed fueling time calls for special attention, because the detour is no longer the only factor considered in making fueling decisions. The model needs to keep both the fueling frequency and detour at low levels.

This paper is organized as follows. In Section 2 a review of the related literature and our contribution is presented. Section 3 describes the VRPAFVFFFT problem and a general Mixed Integer Programming (MIP) formulation for solving the problem. Section 4 gives a hybrid heuristic method which combines Adaptive Large Neighborhood Search (ALNS) with a local search and a MIP model. Section 5 presents the computational results of small, moderate, and large instances. We conclude the paper in Section 6.

2 Literature Review and Contribution

The Vehicle Routing Problem (VRP) is a central problem in transportation (Bektaş et al., 2016). VRP first appeared in a paper by Dantzig and Ramser (1959). It solves a routing problem for trucks to meet the demands from customers located in different places while minimizing the total mileage traveled by the fleet. Since then, VRP has been a hot topic in Operations Research and extended to a number of variants with respect to different real-world constraints. Basic variants of VRP includes Capacitated VRP (CVRP), VRP with Time Windows (VRPTW), Pickup and Delivery Problems (PDP), etc. Nowadays, with increasing concerns on the sustainability of the logistics system, current research on VRP has received more interest on environmental issues, often referred as the Green Vehicle Routing Problem (GVRP). The standard objective function for the traditional VRP is to minimize the total traveling distance, while many of the GVRP papers consider the reduction of pollutants, such as nitrogen oxides (\(N_2O\)), particulate matter (PM), as well as greenhouse gases (GHG). Some papers directly add those items into the objective function, while many more consider fuel consumption related terms (Demir et al., 2014), since fuel consumption can sometimes be used as a surrogate measure for the emissions of air pollutants.

GVRP covers a wide range of topics, and factors which influence fuel consumption are considered in GVRP papers. Based on a survey paper, these factors can be divided into five groups, including vehicle, environment, traffic, driver and operations related categories (Demir et al., 2014). Speed and congestion in
the traffic related category are common factors in many GVRP related papers (Qian and Eglese 2016; Xiao and Konak 2016). Some studies (Bektaş and Laporte 2011; Xiao et al. 2012) also seek to add a payload factor, which belongs to the operations related category, into their models. Various kinds of emission models have been developed to integrate these factors together for fuel consumption or emission calculations. For instance, one of the most widely used emission models is called MEET (Demir et al. 2014), which was first proposed by Hickman et al. (1999). Xiao and Konak (2016) adopt a MEET model to calculate the emissions associated with the arcs between cities based on factors such as speed and payload. Instead of traveling distances, emissions are the new costs for the arcs and a comprehensive MIP formulation is proposed to solve the GVRP. They use CPLEX to solve the MIP and also develop some heuristic methods to deal with large cases. Bektaş and Laporte (2011) present another emissions model to incorporate factors like speed and payload into the cost of emissions, and finally into the routing problem. A MIP formulation with strengthen constraints are given by the authors. Computational results shed light on the tradeoffs between various performance measures.

Speed and congestion as well as payload are very important factors for fuel consumption. However, for alternative fuel vehicles like CNG trucks, some other factors such as fuel stations, may also have a large influence on fuel consumption. Compared with the traditional diesel or gasoline vehicles, the sparse allocation of fuel stations and small fuel tank capacity of alternative fuel vehicles call for an extra attention on the refueling process. Traditional vehicle routing problems assume that the fuel is adequate for covering the whole tour and the vehicles can be refueled any time. However, such an assumption may not hold for CNG trucks. Surveys (Ahmed et al. 2013; Kuby et al. 2013; Ma et al. 2013) show CNG vehicle drivers suffer from issues such as long waiting time at fuel stations and long detour off their least-time routes. Since GVRP for alternative fuel vehicles is a relatively new area, the existing research regarding fuel stations is limited.

So far only few papers have considered fuel stations in their routing problems. In those papers, different models and solution methods are proposed based on the unique features for their specific background. Erdoğan and Miller-Hooks (2012) extend CVRP to GVRP where the routing problem for alternative fuel-powered vehicle fleets is solved. Limited by its tank capacity, an alternative fuel vehicle has to visit one or more times the fuel stations during its tour to serve the customers. Two construction heuristic methods and a customized improvement technique are developed to minimize the total distance traveled by the vehicles. Schneider et al. (2014) present a hybrid heuristic that combines a Variable Neighborhood Search (VNS) algorithm with a Tabu Search (TS) to solve an electric vehicle routing problem. They construct an
initial solution without a guarantee of feasibility and set penalties on the violations of feasibility so that the following search algorithm tends to focus on feasible solutions. The hybrid heuristic is tested on newly designed instances based on the traditional cases, as well as special benchmark cases from other papers like Erdogan and Miller-Hooks (2012). These tests show that hybridization has a positive effect on the solution quality. Yildiz et al. (2016) develop a branch and price approach for the routing and refueling station location problem under a slightly different background by which the demand is given by O-D pairs. Hiermann et al. (2016) propose a more complex model to solve a routing problem for a heterogeneous fleet. They develop a hybrid heuristic which combines an ALNS with an embedded local search and labeling procedure for intensification. The choice of recharging stations in each electric vehicle route is explicitly handled with a labeling procedure. Their method takes advantage of tight time windows, whereas our proposed method does not depend on time windows to reduce the search space. A variant is created by Hiermann et al. (2019), which solves the routing of a mix fleet made up of conventional, plug-in hybrid and electric vehicles. Keskin et al. (2019) consider the capacity limitation in the recharging station and solve the routing problem for electric vehicles with time-dependent queueing times at the stations. Time window constraints are considered in their model. Bruglieri et al. (2019) solve a routing problem with capacitated alternative fuel stations with both arc-based and path-based methods. In their problem, time windows are introduced to simulate the availability of pumps in stations.

In this model, we bring in a fixed fueling time for the fueling procedure, which may represent the expected waiting time at a fuel station. The motivation for the fixed fueling time comes from the real world. Compared with fueling procedures for fossil fuel, users wait for a long time in queues to get their vehicle refueled at alternative fuel stations. For instance, vehicles may wait in the CNG fuel stations for a long time, even though the CNG refueling time itself may only take a few minutes (Ahmed et al., 2013; Khan et al., 2015; U.S. Department of Energy, 2017b). The fixed fueling time can fundamentally change the solution method. With this newly added fixed fueling time, the detour is not the only factor which influences the fueling decision. In some cases, a truck could visit several fuel stations that have a short detour to cover the whole route as opposed to covering the entire route with one visit to a fuel station that has a longer detour. The route with the longer detour to the fuel station may be preferred because of the fewer occurrences of the fixed fueling time. We need to balance the detour and the frequency of visiting fuel stations. This fixed fueling time is different from previous studies on VRP with service times at nodes visited (see for example Errico et al., 2016; Taş et al., 2013). In our problem, the fixed fueling time is independent of the volume of the refueled gas and happens each time a truck visits a fuel station.
Our contributions in this paper are three-fold. From a modeling perspective, we introduce the VR-PAFVFFT to solve the routing problem for alternative fuel vehicles with fixed fueling time. From a solution methodology perspective, we introduce some methods to reduce the search space for an optimal solution. Also we develop an adaptive branch-and-cut algorithm to further improve the computational performance. To deal with instances in realistic size, we extended ALNS, a widely used heuristic method for similar routing problems, by combining with a local search and MIP model. Finally, we provide an extensive set of computational experiments on randomized and actual data sets to provide insights to how the routes are effected when including the extra constraints for AFVs.

3 Problem Statement and Formulation

This section first describes the problem, and then a MIP model is presented to solve the problem. In order to improve the computational performance, variable elimination and valid inequalities are provided and an adaptive branch-and-cut algorithm is proposed.

3.1 Problem Statement

VRPAFVFFT is designed to solve the vehicle routing problem for a distributor that employs a fleet of AFVs, for instance CNG trucks, for daily delivery operations. Customers are distributed in a region, and demands from customers are known ahead of time. We assume the distributor’s depot does not have refueling capability, which means its vehicles have to use the sparsely distributed public fuel stations during their delivery routes. Thus, we need to develop route plans for the fleet which minimizes the total working time while meeting constraints such as fuel tank capacity and load capacity limitations. A typical route for the fleet may start from a departure depot, visit some customers and fuel stations if necessary, and return back to the departure depot at the end of the day.

The main objective for this problem is to minimize the total working time for the vehicles, while satisfying three kinds of constraints which are load capacity, daily working time and fuel tank capacity limitations. We define the working time as the time when the vehicles are away from the depot. It includes traveling time in the routes, fixed fueling time in the stations and variable fueling time which depends on the volume of the fuel that needs to be refilled. To simplify the model, we assume all of the vehicles travel at a constant speed. Furthermore, we assume fuel is also consumed at a constant rate. However, our solution methods are arc based, so we can easily bring in nonlinearity by calculating the time and gas needed for each arc individually.
As previously mentioned, the fixed fueling time could be the expected waiting time for the availability of the pump, which can be as long as hours in the case of CNG vehicles (Kuby et al., 2013). The fixed fueling time may be different from station to station; a station which is more central (close to high population areas) can experience longer waiting times than the other stations. In our model, we assume each station has its own fixed fueling time. The variable fueling time is set to be proportional to the volume of fuel that needs to be refilled. Initially, each vehicles is located at the depot and with a given level of gas left in the tank and each vehicle is required to return back to the depot with a minimal level of gas so that it can at least go to the nearest fuel station on the next day. The initial gas levels for the vehicles are different, so the fleet is heterogeneous. Since the main objective of this paper is to analyze the effect of refueling stops, we ignore the influence of other factors such as speed, congestion and payload and assume the fueling consumption linearly depends on the traveled mileage. We note that although the linear relationship assumption is necessary for our MIP formulation, the heuristic method can easily be extended to include a nonlinear relationship for fuel consumption.

3.2 Mixed Integer Programming Model

An instance network of VRPAFVFFT consists of a set of vertices \( V \) and a set of arcs \( A \). Vertices in set \( V \) are marked from 0 to \( n \). Vertices \( v_0 \) and \( v_n \) are the departure and arrival depots respectively. Though \( v_0 \) and \( v_n \) are two different vertices in our network, in reality they could be located at the same place. The rest of the vertices are divided into two parts, customer vertices \( C \) and fuel station vertices \( F \). Since a vehicle is allowed to visit a station multiple times in its single route, we replicate each station enough times in the network, so that we can keep track of the status (load, gas level, etc) when a vehicle visits a fuel station. An arc \( a_{i,j}^k \) in set \( A \) represents a directed route from vertex \( v_i \) to vertex \( v_j \) for vehicle \( k \) in fleet \( M \). We introduce the index \( k \) for the vehicles because the fleet is heterogeneous since each vehicle has its own unique initial gas level. The arcs in the network include arcs from departure vertex \( v_0 \) to any vertex in \( C \cup F \), from any vertex in \( C \cup F \) to arrival vertex \( v_n \), between any two different vertices in \( C \), and between a vertex in \( C \) and a vertex in \( F \). Note that we also include arcs between vertices in \( F \), namely we allow successive visits to the fuel stations. We introduce more notation for the MIP formulation. Let \( V_i^+ \) and \( V_i^- \) be the sets of vertices which have arcs from and to vertex \( v_i \), respectively. Overall, the fleet has \( m \) vehicles and all of them need to depart from vertex \( v_0 \) and arrive at vertex \( v_n \) at the end of the day.

In the MIP model, binary variable \( x_{i,j}^k \) is equal to 1 if arc \( a_{i,j}^k \) is traveled by vehicle \( k \) in the solution and 0 otherwise. Continuous variables \( t_i^k \), \( g_i^k \) and \( c_i^k \) keep track of the working time from the departure depot,
remaining gas level and current delivered load of vehicle $k$ at vertex $v_i$, respectively.

Let constant $R_f$ represent the fixed fueling time for a vehicle’s single visit to a fuel station $f$. Constants $T$ and $G$ are the limitations for daily working time (in minutes) and fuel tank capacity (in gallons). Constant $L$ is the limitation of a vehicle’s cargo load. We assume all vehicles’ daily working time, fuel tank capacity and cargo load are identical. Namely $T$, $G$ and $L$ are the same for all vehicles. Let $\gamma_1$, $\gamma_2$ and $\gamma_3$ be the speed of the vehicles (miles/minute), fuel consumption rate (miles/gallon) and refueling rate (gallons/minute) respectively. $\delta_{i,j}$ is the distance from vertex $v_i$ to vertex $v_j$. $\sigma_i$ is the load for vertex $v_i$. It is the customer’s demand if the vertex is a customer vertex and 0 otherwise. All of the customers’ demands are known in our model. Constants $\tau_k$ (in gallons) are the initial gas level for vehicles $k$ and the initial gas level can be different from vehicle to vehicle. For each vehicle, $\tau_k$ should be set to be a number between $\min_{j \in F} \left( \frac{\delta_{i,j}}{\gamma_2} \right)$ and $G$ since within this range it guarantees the vehicle has enough fuel to go to the nearest fuel station at the beginning of the period.
3.2.1 Basic Formulation

Based on Cordeau [2006]'s work, we develop an arc-based formulation to solve the problem. The basic MIP model for VRPAFVFFT is as follows.

\[
\begin{align*}
\text{min} & \quad \sum_{k \in M} t_n^k \\
\text{s.t.} & \quad \sum_{i \in V^-_j} x_{i,j}^k - \sum_{i \in V^+_j} x_{j,i}^k = 0 \quad \forall j \in C \cup F, k \in M \\
& \quad \sum_{i \in V_j^-} \sum_{k \in M} x_{i,j}^k = 1 \quad \forall j \in C \\
& \quad \sum_{i \in V^-_j} x_{i,j}^k \leq 1, \quad \forall k \in M \\
& \quad \sum_{i \in V^+_j} x_{i,j}^k \leq 1, \quad \forall j \in F, k \in M \\
& \quad t_i^k + M (x_{i,j}^k - 1) + \frac{\delta_{i,j}}{\gamma_1} \leq t_j^k, \quad \forall i \in \{0\} \cup C, j \in V^+_i, k \in M \\
& \quad t_i^k + M (x_{i,j}^k - 1) + \frac{\delta_{i,j}}{\gamma_1} + \frac{G - g_i^k}{\gamma_3} + R_i \leq t_j^k, \quad \forall i \in F, j \in V^-_i, k \in M \\
& \quad 0 \leq t_i^k \leq T, \quad \forall i \in \mathcal{V}, k \in M \\
& \quad g_i^k + M (1 - x_{i,j}^k) - \frac{\delta_{i,j}}{\gamma_2} \geq g_j^k, \quad \forall i \in \{0\} \cup C, j \in V^+_i, k \in M \\
& \quad M (1 - x_{i,j}^k) + G - \frac{\delta_{i,j}}{\gamma_2} \geq g_j^k, \quad \forall i \in F, j \in V^+_i, k \in M \\
& \quad 0 \leq g_i^k \leq G, \quad \forall i \in \mathcal{V}, k \in M \\
& \quad g_0^k = \tau_k, \quad \forall k \in M \\
& \quad g_n^k \geq \min_{j \in F} \left( \frac{\delta_{n,j}}{\gamma_2} \right), \quad \forall k \in M \\
& \quad c_i^k + M (1 - x_{i,j}^k) - \sigma_i \geq c_j^k, \quad \forall i \in \{0\} \cup C \cup F, j \in V^+_i, k \in M \\
& \quad 0 \leq c_i^k \leq L, \quad \forall i \in \mathcal{V}, k \in M \\
& \quad x_{i,j}^k \in \{0,1\}, \quad \forall i \in \mathcal{V}, j \in \mathcal{V}^+_i, k \in M
\end{align*}
\]

Objective function \((1)\) is to minimize the total working time, which includes traveling time, fixed fueling time and variable fueling time. Constraints \((2)\) and \((3)\) are the flow conservation equations that ensure that the solution defines a route plan for the fleet to visit every customer. Constraint \((4)\) restricts that each vehicle can be used at most once. Constraint \((5)\) guarantees each replicate of the fuel station can
be visited by the same vehicle at most once. Constraints (6) to (8) are working time related constraints. The daily working time increases due to the traveling time only if the vehicle travels through the regular arcs, which is represented in constraint (6). It increases due to the traveling time, variable fueling time and fixed fueling time if it travels through arcs adjacent to the fuel stations, which is shown in constraints (7). Constraint (8) bounds the daily working time between 0 and $T$. The fuel tank capacity constraint is covered by constraints (9) to (13). Constraint (9) decreases the remaining gas level if a vehicle travels through a regular arc. Constraint (10) means if a vehicle travels through an arc starting from a fuel station, the remaining gas level will be set to the full gas level minus the gas used for that arc, namely the vehicle is fully refueled at the fuel station. The remaining gas level boundary, initial gas level and final gas level are set in constraints (11), (12) and (13) respectively. In constraint (13), we require that a vehicle returns back to the arrival node with enough gas to go to the nearest fuel station if necessary. Constraints (14) and (15) guarantee the demand for each customer can be fulfilled by the assigned vehicle. Constraint (16) is the domain constraint.

### 3.2.2 Preprocessing and Valid Inequalities

Before solving the MIP formulation, some variables are eliminated based on the feasibility of the arcs, and parameters like big $M$ are appropriately selected; and valid inequalities which are redundant but can strengthen the formulation are added to the model.

**Variable elimination.** Variables are eliminated based on different feasibility rules. The first elimination rule is based on the tank capacity limitation. Variable $x_{i,k}^{i,j}$ is eliminated if the corresponding arc distance $\delta_{i,j}$ satisfies one of the conditions (17) to (20).

\[
\delta_{i,j} > \gamma_2 G \\
\min_{i' \in V_i^-} \delta_{i',i} + \delta_{i,j} + \min_{j' \in V_j^+} \delta_{j,j'} > \gamma_2 G, \quad i \in C, j \in V_i^+ \cap C \\
\delta_{i,j} + \min_{j' \in V_j^+} \delta_{j,j'} > \gamma_2 G, \quad i \in F, j \in V_i^+ \\
\min_{i' \in V_i^-} \delta_{i',i} + \delta_{i,j} > \gamma_2 G, \quad i \in C, j \in V_i^+ \cap F
\]  

(17)  
(18)  
(19)  
(20)

Condition (17) is intuitive. A vehicle can not travel on an arc if it is more than the tank capacity limitation $G$. Condition (18) is a strengthened version of condition (17), which means an arc between two customers is not reachable if a vehicle can not travel from any other place to the arc and go to any other place. Similarly,
in conditions (19) and (20), an arc away from a fuel station, or to a fuel station is not reachable if the
minimal distance of an adjacent arc combined with this arc is larger than the tank capacity limitation.

The second rule is for symmetry elimination. The first symmetry elimination condition (21) is modified
from Cortés et al. (2010). Note that this condition works only if the fleet is homogeneous, namely all of the
vehicles are identical, including the initial gas level. If the fleet is homogeneous, two solutions are identical if
we can transfer from one to the other by changing the numbering of the vehicles. Condition (21) implies that
the customer vertices in $\mathbb{C}$ are visited in order by each one of the vehicles. For instance, for customer 2, the
demand can only be fulfilled by vehicle 1 or 2. Based on the same principle, we come up with conditions (22)
and (23), which can be used in the heterogeneous fleet case. For a single truck, the replicates of a same fuel
station are identical. Conditions (22) and (23) imply that replicates of a fuel stations are visited in order.
In these two equations, we introduce notation $i$ to represent that the corresponding vertex in the network
is the $i$th replicate of a fuel station. An instance of these two constraints is a vehicle may only go to the
first and second replicates of the fuel stations immediately after or before visiting customer 2’s vertex.

$$x_{i,j}^k = 0, \quad \forall i \in \mathbb{C}, j \in \mathbb{V}^+_i, k \in \mathbb{M}, k > i \quad (21)$$

$$x_{i,j}^k = 0, \quad \forall i \in \mathbb{C}, j \in \mathbb{V}^+_i \cap \mathbb{F}, (j) > i, k \in \mathbb{M} \quad (22)$$

$$x_{i,j}^k = 0, \quad \forall j \in \mathbb{C}, i \in \mathbb{V}^-_j \cap \mathbb{F}, (i) > j, k \in \mathbb{M} \quad (23)$$

The third elimination rule is induced by the domination of the fuel stations. We say a fuel station vertex
$v_{f_1}$ dominates another $v_{f_2}$ between a pair of customer vertices $v_i$ and $v_j$, if $\delta_{i,f_1} \leq \delta_{i,f_2}$ and $\delta_{f_1,j} \leq \delta_{f_2,j}$ are
satisfied. A fuel station vertex $v_{f_2}$ can never be an option between $v_i$ and $v_j$ in the optimal solution if it is
dominated by any other fuel station. Variables $x_{i,f_2}$ is eliminated if the fuel station vertex $v_{f_2}$ is dominated
by other fuel stations in any pair between $v_i$ and a vertex successive to $v_i$, namely condition (24) holds.

$$\delta_{i,f_1} \leq \delta_{i,f_2}, \delta_{f_1,j} \leq \delta_{f_2,j}, \quad i \in \mathbb{C}, f_1, f_2 \in \mathbb{F}, f_1 \neq f_2, \forall j \in \mathbb{V}^+_{f_1} \cap \mathbb{V}^+_{f_2} \quad (24)$$

The fourth elimination rule is induced by the remaining gas level after visiting a fuel station. Variable $x_{i,j}^k$
representing an arc from customer vertex $v_i$ to fuel station vertex $v_j$ by vehicle $k$ is eliminated if condition (25)
holds.

$$\frac{\delta_{i,j} - \delta_{i,j'}}{\gamma_2} > G - \frac{\delta_{j,j'}}{\gamma_2}, \quad \forall j' \in \mathbb{V}^+_j \quad (25)$$
A vehicle can travel from vertex \( v_i \) to \( v_j \) only if the remaining gas can cover distance \( \delta_{i,j} \). The left-hand side of condition (25) is the minimal remaining gas at vertex \( v_j \), if the vehicle goes directly from vertex \( v_i \) to \( v_j \), while the right-hand side is the remaining gas if the vehicle goes to the fuel station. A visit from customer vertex \( v_i \) to a fuel station vertex \( v_j \) is not necessary if the remaining gas level after the visit is always lower no matter where the vehicle goes.

The fifth elimination rule is also related to the visiting of a fuel station. A visit from customer vertex \( v_i \) to a fuel station vertex \( v_j \) is impossible if the minimal tour including this arc is larger than the daily working time limitation. The right-hand side of condition (26) shows the minimal tour.

\[
\delta_{0,i} + \delta_{i,j} + \delta_{j,0} > \gamma_1(T - R_j)
\]  

(26)

The last elimination rule concerns the load capacity limitation. \( x_{i,j}^k \) is eliminated if condition (27) is satisfied, namely the demands \( \sigma_i \) and \( \sigma_j \) are more than a vehicle’s load capacity.

\[
\sigma_i + \sigma_j > L
\]  

(27)

**Parameter setting** Parameters like big \( M \) and the number of replicates of each fuel station are explicitly handled. In theory the big \( M \) in constraints (6), (7), (9), (10) and (14) can be any number as long as it is sufficiently large. However, an arbitrary big \( M \) may bring in some numerical problems and weaken the bound from the linear relaxation of the MIP. We need to choose \( M \) as small as possible to accelerate the computation speed. For constraints (6) and (7) the minimal \( M \) should satisfy that even if \( t_{i,j}^k \) reaches its maximal possible value, the minimal possible value of \( t_{i,j}^j \) will not be influenced when \( x_{i,j}^k = 0 \). \( M_1 \) and \( M_2 \) in equations (28) and (29) are the detailed representation of \( M \) for constraints (6) and (7). Similarly, big \( M \) in constraints (9) and (10) are given by \( M_3 \) and \( M_4 \) in equations (30) and (31). For constraint (14) big \( M \)
is simply set to be 1.

\[ M_1 = \left( T - \frac{\delta_{i,n}}{\gamma_1} \right) - \frac{\delta_{b,j}}{\gamma_1} + \frac{\delta_{i,j}}{\gamma_1} \]  

\[ M_2 = \left( T - \frac{\delta_{i,n}}{\gamma_1} \right) - \frac{\delta_{b,j}}{\gamma_1} + \frac{\delta_{i,j}}{\gamma_1} + \frac{G}{\gamma_3} + R_i \]  

\[ M_3 = \max\{g_1, g_2\} - \min_{\nu^i \in \mathcal{V}_i^+ \cup \mathcal{F}} \frac{\delta_{i,\nu}}{\gamma_2} + \frac{\delta_{i,j}}{\gamma_2}, \]  

where \( g_1 = \tau - \frac{\delta_{b,j}}{\gamma_2}, \) \( g_2 = G - \min_{j' \in \mathcal{V}_j \cup \mathcal{F}} \frac{\delta_{j',j}}{\gamma_2} \)  

\[ M_4 = \max\{g_1, g_2\} - G + \frac{\delta_{i,j}}{\gamma_2}, \]  

where \( g_1 = \tau - \frac{\delta_{b,j}}{\gamma_2}, \) \( g_2 = G - \min_{j' \in \mathcal{V}_j \cup \mathcal{F}} \frac{\delta_{j',j}}{\gamma_2} \)  

As we stated in the previous section, we replicate the fuel stations in case a vehicle needs to visit a fuel station multiple times. Too many replicates will increase the number of variables dramatically. The upper bound on the number of times a vehicle visits a fuel station is given by \( \left\lceil \frac{2T\gamma_1}{\gamma_2 G} \right\rceil + 1. \)

![Figure 1: An example of a vehicle’s multiple visits to a fuel station](image)

Figure 1 shows an example of a vehicle’s multiple visits to a fuel station. A vehicle starts from a depot, goes through a fuel station, visits some customers in the subtours and goes back to the depot. We claim that only one subtour’s distance can be less than \( \frac{\gamma_2 G}{2}. \) Otherwise, the subtours can be combined together with less distance because of the triangle inequality with one less visit to a fuel station. Thus the total number of times a vehicle visits a fuel station is bounded by \( \left\lceil \frac{2T\gamma_1}{\gamma_2 G} \right\rceil + 1. \)

**Valid inequalities** Some valid inequalities are added to the model to strengthen the linear relaxation of the problem. The valid inequalities we add are from the tank capacity limitation. If the total distance of two successive arcs is larger than \( \gamma_2 G, \) a vehicle can not travel through both of the arcs. Suppose the
vertices are \(v_{i_1}, v_{i_2}\) and \(v_{i_3}\), inequality (32) is added to the formulation if \(\delta_{i_1,i_2} + \delta_{i_2,i_3} > \gamma_2 G\).

\[
x^{k}_{i_1,i_2} + x^{k}_{i_2,i_3} \leq 1, \quad \forall k \in M
\]  

(32)

### 3.2.3 Adaptive Cut Generation

The branch-and-cut algorithm is a popular algorithm for solving combinatorial optimization problems. We generate the initial pool of cuts based on the load capacity limitation, and the dominance relationship in the fueling process as well as the regular subtour elimination from Cordeau (2006).

The first group of cuts is from the load capacity limitation. If a subset \(C'\) of the customers’ cargo load is more than a vehicle’s load capacity, those subsets of customers cannot be served by a vehicle. We select those subsets whose cardinality is 3, namely \(|C'| = 3\), and add inequality (33) into the initial pool of cuts for all such \(C'\).

\[
\sum_{i \in C'} \sum_{j \in C', i \neq j} x^{k}_{i,j} \leq |C'| - 2 \quad \forall k \in M
\]  

(33)

The second group of cuts is with the remaining gas level after visiting a fuel station, which is similar to inequality (25). Let \(v_{i_1}, v_{i_2}\) represent two customer vertices, and \(v_j\) represents a fuel station vertex. Inequality (34) is added into the initial pool of cuts if \(\gamma_2 (\delta_{i_1,j} - \delta_{i_1,i_2}) \geq G - \gamma_2 \delta_{j,i_2}\).

\[
x^{k}_{i_1,j} + x^{k}_{j,i_2} \leq 1, \quad \forall k \in M
\]  

(34)

We generate the last group of cuts based on the subtour elimination inequality (35), where \(C'\) is a subset of the customers.

\[
\sum_{i,j \in C', i \neq j} x^{k}_{i,j} \leq |C'| - 1
\]  

(35)

Cuts are added to the pool if the cardinality of \(C'\) is 2, and the corresponding two vertices do not appear in any set from (33).

Preliminary experiments show that a significant amount of time is spent in searching for violated inequalities in the pool of cuts. In many cases, the searching time is even comparable to the regular solving time. We propose an adaptive cut generation to improve the searching efficiency. The idea of adaptive cut generation is a group of cuts will be checked with higher probability during the branch-and-bound process if this group
of cuts are likely to be violated. We set \( p_{i,t} \) be the probability of checking group \( i \)'s cuts in iteration \( t \). In our model, \( i = 1, 2, 3 \), refer to cut the groups from the load capacity limitation, remaining gas level and subtour elimination respectively. Initially \( p_{i,0} = 1, i = 1, 2, 3 \). We define \( \text{lhs}_{i} \) and \( \text{rhs}_{i}, i = 1, 2, 3 \) as the left-hand and right-hand side of inequalities (33), (34) and (35), respectively. Take one cut \( j \) in inequality (33) where \( i = 1 \) as an example,

\[
\text{lhs}_{i,j}^j = \sum_{i \in C} \sum_{j \in C', i \neq j} x^{k}_{i,j} \quad (36)
\]

\[
\text{rhs}_{i}^j = |C'| - 2 \quad (37)
\]

Let \( \text{lhs}_{i,t}^j \) be the value of \( \text{lhs}_{i,t}^j \) in iteration \( t \) by substituting \( x^{k}_{i,j} \) with its value in the iteration. Since the right-hand side of each group of cuts has the same constant value, we simply use \( \text{rhs}_{i}, i = 1, 2, 3 \) to denote the right-hand side values. We update \( p_{i,t} \) as follows.

\[
p_{i,t+1} = (1 - r)p_{i,t} + r \prod_{j} \frac{\text{lhs}_{i,t}^j}{\text{rhs}_{i}} \quad (38)
\]

where \( r \) is a constant number and we set it to 0.01 for our problem. From equation (38) we know the score of a cut is defined by the \( \frac{\text{lhs}_{i,t}^j}{\text{rhs}_{i}} \) part. It is bigger than 1 if the cut is violated. Thus the corresponding group of cuts has a higher probability to be checked in the next iteration if the cut is violated in the current iteration.

4 Heuristic Solution Method

ALNS is an efficient heuristic method for solving the classical vehicle routing problem. It is proposed by Ropke and Pisinger (2006) from Large Neighborhood Search (LNS), which is first introduced by Shaw (1998) for the routing problem. Hiermann et al. (2016) extend it by embedding a local search and labeling procedure to solve a vehicle routing problem with time windows and recharging stations. In this section, we propose a new hybrid heuristic algorithm that combines an ALNS with a MIP model to solve the routing problem with fuel stations. Our new method has better performance when the search space is large. In our method, the search is conducted within the feasible region. This procedure has an advantage that whenever the method is stopped, we will always get a feasible solution for the fleet.

We add one more assumption in our model for the heuristic method. A vehicle can visit at most one fuel station between two adjacent customers on its route. In some very rare cases, the optimal route for a
vehicle may contain two successive visits of fuel stations. For instance, two customers might be very far away from each other such that a vehicle has to visit multiple fuel stations in order to travel from one customer to another. Since our problem solves the daily routing problem for a fleet performing delivery operations within a city and we assume the tank capacity is high enough to travel from any fuel station in the region to any demand point in the region, we ignore such a situation in the heuristic solution approach. Based on the settings in the problem statement, we introduce some more notations. Let \( r_k \) be the route for vehicle \( k \). \( r_k(i) \) is the \( i \)th node in route \( k \). Figure 2 shows an example of route \( k \).

From the problem statement in the previous section, the first node \( r_k(0) \) and the last node \( r_k(j) \) are the depot vertices \( v_0 \) and \( v_n \). Within the route, some nodes, for instance \( r_k(i+1) \), may be fuel stations. Let \( \text{Time}(r_k) \) be the total working time for route \( k \) and \( \text{Load}(r_k) \) be the total load for route \( k \). Let \( \text{Time}(r_k(i)) \) and \( \text{Load}(r_k(i)) \) be the working time and cargo load from the start depot until the current node \( r_k(i) \). Let \( \text{Fuel}(r_k(i)) \) be the remaining gas level at node \( r_k(i) \). Note \( \text{Time}(r_k) \) and \( \text{Load}(r_k) \) are bounded by \( T \) and \( L \) respectively. \( \text{Fuel}(r_k(i)) \) is bounded by \( G \) and \( \text{Fuel}(r_k(0)) = \tau, \text{Fuel}(r_k(j)) \geq \min_{f \in F} \left( \frac{\delta_n f \gamma^2}{2} \right) \). All of these boundaries are identical with the ones in the MIP formulation in Section 3.2.

In the heuristic solution method, between any two successive nodes \( i \) and \( i+1 \), only the fuel stations with the smallest total detour are considered during the insertion. Let \( \delta^+_{r_k(i),r_k(i+1)} \) denote the distance from a customer node \( r_k(i) \) to the fuel station and \( \delta^-_{r_k(i),r_k(i+1)} \) denote distance from the fuel station to a customer node \( r_k(i+1) \). Let \( c^k \) be the cost of vehicle \( k \) and \( C \) be the fleet’s total cost, then equation (39) is the objective function of the hybrid heuristic algorithm. Thus the objective is identical with the MIP model.

\[
C = \sum_{k \in M} c^k = \sum_{k \in M} \text{Time}(r_k) \quad (39)
\]

4.1 Initial Feasible Solution Construction

The construction of an initial feasible solution is modified from Ropke and Pisinger’s parallel insertion heuristic (Ropke and Pisinger 2006). Figure 3 shows the general steps for the construction process. We start from an empty fleet. In each loop, we pick a customer’s demand which is not inserted to the fleet yet, go through all positions in the current fleet for the customer’s demand to check if the route plan is still valid.
after the insertion, select the position with the smallest additional traveling distance or add one more vehicle
if there is not a valid insertion position for the current customer’s demand, insert the customer’s demand
into the fleet and repair the influenced route if necessary. We repeat the procedures until all demands are
inserted. The following section gives a more detailed explanation on valid insertion, additional traveling
distance, customer’s demand insertion and route repair.

Figure 3: Steps of constructing an initial feasible solution

Suppose customer vertex \( v_i \) is going to be inserted into vehicle \( k \)'s route, after the \( j \)th node. A valid
insertion means it meets four constraints, which are working time, fuel tank capacity, load capacity and
location constraints. An insertion is also valid if it only violates the fuel tank capacity limitation, which
can be simply repaired by adding a fuel station in front of the inserted customer while keeping all the other
constraints valid. Details about the four constraints are listed in the following equations (40) to (45).

**Working time constraint.** Equation (40) shows the criterion for the working time constraint. The daily
working time can not exceed \( T \).

\[
\text{Time} \left( r^k \right) + \frac{\delta_{r^k_{(j+1)*}} + \delta_{r^k_{(j+1)}} - \delta_{r^k_{(j+1)}, r^k_{(j+1)*}}}{\gamma_1} \leq T
\]

**Fuel tank capacity constraint.** For the fuel tank capacity constraint, equations (41) to (43) show its
criteria. Equation (41) is easy to understand. It ensures the vehicle has enough fuel to go to the fuel station.
In equation (42), \( \text{Fuel} \left( r^k_{(j+1)*} \right) \) is the gas level at node \( r^k_{(j+1)*} \), where node \( r^k_{(j+1)*} \) is the first fuel station
after node $r_{(j+1)^*}$, or the last node if there is no fuel station after it. The quantity in the left-hand side of equation (42) is the gas needed to cover the range between the newly inserted fuel station and the nearest following fuel station, or the last node $r_{(j+1)^*}$. This quantity should be no bigger than the tank capacity $G$. It eliminates a special case in which the distance from the fuel station back to vertex $v_i$ is so large that inserting a fuel station into the route still can not make it feasible. Equation (43) ensures the vehicle still meets the daily working time constraint if it needs to refuel. The working time changes due to the extra traveling time and extra fueling time, and the extra fueling time includes changes in the variable fueling time and the fixed fueling time. Note that we use $R_f$ to denote the corresponding fuel station’s fixed fueling time.

Load capacity constraint. The load capacity constraint is explained in Equation (44). The load of vehicle $k$ can not be more than $L$.

Location constraint. The last constraint for a valid insertion is the location constraints. Given the assumption that fuel stations can not be visited successively, node $r_{(j)}$ can not be a fuel station vertex.

Let $\Delta d^k$ denote the change in distance incurred by a valid insertion of customer vertex $v_i$ into route $k$ at the position that increases the distance for vehicle $k$ the least. The change, or increase in distance, is simply

$$\Delta f = \frac{G - Fuel \left( r_{(j)} \right) - \gamma^{-1}_2 \delta^{+}_{r_{(j)},i}}{\gamma_3} - \frac{G - \gamma_2^{-1} \left( \delta^{-}_{r_{(j)},i} + \delta_{i,r_{(j+1)^*}} \right) - Fuel \left( r_{(j+1)^*} \right)}{\gamma_3} + R_f$$ (43)
measured by $\delta_{r_{(j+)}^{k},r_{(j)}^{k},i} + \delta_{i,r_{(j+1)}^{k},r_{(j)}^{k}}$. Initially $\Delta d^k_1$ is set to be $+\infty$ and $\Delta d^k_1$ is updated by equation (46).

$$\Delta d^k_1 = \min_{j \in J^k_i} \left( \delta_{r_{(j)}^{k},i} + \delta_{i,r_{(j+1)}^{k}} - \delta_{r_{(j)}^{k},r_{(j+1)}^{k}} \right)$$  \hspace{1cm} (46)

where $J^k_i$ is the set of valid insertion positions for vehicle $k$ for customer vertex $v_i$. Finally we choose vehicle $k$ that minimizes $\Delta d^k_1$ and insert customer vertex $v_i$ at the minimum cost position in that vehicle. If $\Delta d^k_1 = +\infty$ for all $k$, there is no valid insertion for customer vertex $v_i$ and we need to use another vehicle route.

After inserting customer vertex $v_i$ into vehicle $k$, we need to process an inspection on the vehicle’s fuel tank capacity constraint. If there exists a node $r_{(j)}^{k}$ whose $Fuel(r_{(j)}^{k})$ is less than 0, an insertion of the fuel station is required. Though equations (41) to (43) ensure that inserting a fuel station with minimum detour in front of $v_i$ will make the route feasible again, we still want to find a feasible position $j$ so that $Fuel(r_{(j)}^{k})$ is bigger than 0 while as small as possible. In reality, this strategy means that we try to delay the fueling as much as possible. The vehicle will keep working until its fuel tank is almost empty. Intuitively, this delay will minimize the number of visits to the fuel stations. The search of position $j$ starting from the first node $r_{(m)}^{k}$ in route $r^k$ whose $Fuel(r_{(m)}^{k})$ is less than 0. We set $j = m$ and if two adjacent nodes $r_{(j-1)}^{k}$ and $r_{(j)}^{k}$ satisfy equations (41) to (43), we find feasible position $j$. Otherwise we set $j = j - 1$ and check again, until we find a feasible position. In the worst case, we will stop at the position where we just inserted the customer vertex $v_i$.

The insertion procedure stops when all customers are inserted into the fleet.

### 4.2 Adaptive Large Neighborhood Search

The ALNS is adapted from Hiermann et al. (2016). Different than this study, our search is conducted only within the feasible region. The parameters are learnt from Ropke and Pisinger (2006) and tuned based on our instances. Figure 4 shows the steps of one iteration of the ALNS. We first do local search to intensify the search in each iteration. The next step is the destroy and repair procedures from the general methodology of ALNS. We also try to make the used fleet size as small as possible in the following step. The last step uses a MIP model to find the optimal fueling plan for each route.
4.2.1 Local Search

We introduce some well-known local search methods like shift, swap, 2-opt and 2-opt* to improve the solution quality in each search iteration. These methods are widely used in various kinds of heuristic methods for vehicle routing problems (Chen and Wu, 2006; Hiermann et al., 2016; Montané and Galvao, 2006). In our paper, graphs from Hiermann et al. (2016) are used for illustration purposes.

**Shift.** Figure 5a shows the shift movement. We randomly select a node, for instance the $i$th node from route $k$, and insert it into another randomly selected route $k'$.

**Switch.** In the switch movement, two nodes $i$ and $j$ are randomly picked from two routes $k$ and $k'$, and switch positions, as shown in Figure 5b.

**2-Opt.** In this movement, we first randomly select two positions $i$ and $j$, reverse the order of nodes between $i$ and $j$, and insert into the original position. Figure 5c shows this movement.
2-Opt*. 2-Opt* movement, also called 2-Exchange, is a movement within two routes. Figure 5d is an example of a 2-Opt* movement. Two routes $k$ and $k'$ exchange the rest of the nodes at positions $i$ and $j$.

To select the local search methods to use, we adapt the weighted roulette wheel selection method and adaptive weight adjustment from Ropke and Pisinger (2006). Similar to their work, we define a segment as a number of iterations of the local search; here we define a segment as 10,000 iterations. Let $w_{i,t}^s$ represent the weight for local search method $i$ at segment $t$. We select method $i$ with probability given by equation (47) at segment $t$. The weights are updated via equation (48) and initial weights $w_{i,0}^s, i = 1, 2, 3, 4$ for the four methods are equal.

$$\frac{w_{i,t}^s}{\sum_{j=1}^{4} w_{j,t}^s} \quad i = 1, 2, 3, 4 \quad (47)$$

$$w_{i,t+1}^s = w_{i,t}^s(1 - r^s) + r^s \frac{\pi_{i,t}}{\theta_{i,t}} \quad i = 1, 2, 3, 4 \quad (48)$$

In equation (48), $r^s$ is a constant number, $\pi_{i,t}$ is the score of method $i$ obtained during segment $t$; here we set it to be the decrease in the objective function, and $\theta_{i,t}$ is the number of times we have attempted to use method $i$ during segment $t$. In our experiments, we select $w_{i,0}^s = 100$ and $r^s = 0.5$.

Note that the search is conducted within the feasible region. The first acceptance criterion is that the new routes after the local search should still be feasible. The feasibility check procedures are similar to equations (40) to (43) in Section 4.1. The second criterion is from the Simulated Annealing method, which is also used by Ropke and Pisinger (2006). To avoid getting trapped in a local minimum, an inferior solution may also be accepted with some probability. A solution will be accepted with probability

$$e^{-(C' - C)/T^s} \quad (49)$$

where $C'$ is the cost of the new routes and $C$ is the original one. $T^s$ is the temperature and starts with an initial value $T_{0}^s$. In each iteration $T^s$ is updated via $T^s = \alpha^s * T^s$, where $0 < \alpha^s < 1$. We select $T^s = 10$ and $\alpha^s = 0.9999$ for our experiments.

Since local search can be conducted very fast, we run $\Phi$ iterations in every ALNS iteration. If after a local search iteration a route is empty, or has fuel stations only, it will be deleted. Based on our test, we set $\Phi = 100,000$ in the experiments.
4.2.2 Destroy and Repair Operators

The destroy and repair operators we introduce in this section are mainly from [Ropke and Pisinger 2006]. We get a new solution by removing some nodes using the destroy operator, followed by the repair operator. These operators extend the search space by increasing the search neighborhood. We implement three destroy operators and two repair operators. Notice only the nodes representing customer vertices will be considered in the destroy and following repair operators. We want to generate a feasible solution whenever the algorithm stops. A destroy of a fuel station might make it hard to generate a feasible solution during the repair procedure.

**Random destroy operator.** A random destroy operator is the simplest operator. Every node representing a customer vertex is chosen with equal probability.

**Related destroy operator.** The idea of the related destroy operator is to remove similar nodes together, and reinsert them into routes again. Intuitively if the removed nodes are far away from each other, the solution after the repair operators may still be similar to the original one, or even worse. The relatedness $\text{Related}(i, j)$ for two vertices $v_i$ and $v_j$ is from [Shaw 1998].

$$\text{Related}(i, j) = \frac{1}{\delta'_{i,j} + V_{i,j}}$$  \hspace{1cm} (50)

In equation (50), $\delta'_{i,j}$ is the normalized distance from vertex $v_i$ to vertex $v_j$ and $\delta'_{i,j} = \frac{\delta_{i,j}}{\max \delta_{i,j}}$, where $\delta_{i,j}$ is the distance between vertex $v_i$ and vertex $v_j$ and $\max \delta_{i,j}$ is the maximal one for all of the distances. $V_{i,j}$ is an indicator variable. It is 1 if vertex $v_i$ and vertex $v_j$ are in the same route and 0 otherwise. When using the related destroy operator, the first node is randomly selected with the random destroy operator, followed by 4 nodes, selected by the relatedness of the first node. A higher relatedness value means the higher probability the node will be chosen.

**Worst destroy operator** The idea of the worst destroy operator is if the node leads to a high detour in its route, it is likely to be misplaced and should be removed. This operator is used by [Ropke and Pisinger 2006]. First we calculate the detour $\text{Detour}(i)$ of the $i$th node in route $k$ via equation (51).

$$\text{Detour}(i) = \delta^k_{(i-1),i} + \delta^k_{(i+1),i} - \delta^k_{(i-1),i} - \delta^k_{(i),i+1}$$  \hspace{1cm} (51)

Then we choose the node to be removed based on $\text{Detour}(i)$. A higher detour value means the higher probability the node will be chosen.
Cheapest route repair operator. This repair operator is exactly the same as the insertion method in Section 4.1.

Cheapest customer repair operator. In this operator, we first calculate the cheapest insertion cost for every customer removed by the destroy operators. The procedure is the same as the cheapest route repair operator and is given in Section 4.1. Next we choose the customer with the lowest overall cost and insert it into the corresponding route using the same weighted roulette selection method as in Section 4.1. The parameters in equations (47) and (48) for the destroy operator are $w_{i,t}^d, i = 1, 2, 3$ and $\alpha^d$. The parameters for the repair operators are $w_{i,0}^r, i = 1, 2$ and $\alpha^r$. The segment is redefined as 100 iterations for both the destroy and repair operators. After tuning the parameters, the initial values for $w_{i,0}^d, i = 1, 2, 3$ and $w_{i,0}^r, i = 1, 2$ are all 100, and $\alpha^d = \alpha^r = 0.8$.

Every time before using the destroy and repair operators, we first decide how many nodes are going to be removed. In our case, we randomly generate a number within $\lambda_1 |C|$ and $\lambda_2 |C|$, where $\lambda_1$ and $\lambda_2$ are constants and $0 < \lambda_1 < \lambda_2 < 1$ and $|C|$ is the total number of customers. Next we choose one destroy and one repair operator for this iteration. The selection method is similar to the weighted roulette wheel selection method and adaptive weight adjustment described in Section 4.2.1. If a route after the destroy operator is empty, or has fuel stations only, it will be deleted. In our experiments, $\lambda_1 = 0.05$ and $\lambda_2 = 0.15$.

4.2.3 Fleet Size Minimization

A lower bound for the fleet size can be calculated from the customers’ cargo load. Let $S^*$ be the lower bound.

$$S^* = \left\lceil \sum_{i \in C} \sigma_i \right\rceil (52)$$

If the current fleet size $S$ is bigger than $S^*$, we select $S - S^*$ routes with the smallest number of customers, delete these routes and put the customers into the request pool again. We insert those customers into the fleet via repeating the construction procedure in Section 4.1. The new fleet will be accepted if its size $S'$ is smaller than $S$, otherwise we keep the original one. Though the new fleet size is usually smaller, the total cost $C$ is likely to be larger because of the relatively narrow view on the insertion procedure.

Preliminary experiments show the fleet size minimization procedure will dramatically slow down the computing speed and increase the total cost. Thus, we set the frequency of this procedure to be very low. We run it every $\Phi_2$ ALNS iterations. We set $\Phi_2$ to be 200 in our experiments so that the fleet size is usually minimized, while the computing speed is not affected too much.
4.2.4 Fuel Station Optimization

Hiermann et al. (2016) also add the recharging stations into their model and implement a special labeling algorithm to explicitly handle these stations. The idea of the labeling algorithm is to use a specific design label to denote all of the possible recharging plans and pick the one with the lowest total cost. The success of the labeling algorithm heavily depends on the elimination of the search space, namely the so-called dominance checks in their paper. In their model, each customer has a relatively tight time window so the search space is significantly reduced. In our setting, we do not have any time window constraints which makes the dominance checks to be very weak. Without time window constraints, a detour to a fuel station between two customers has a disadvantage on longer total traveling distance, but has likely an advantage on having more gas left, which may save future detours to the fuel stations. This relationship holds for every pair of successively visited customers, so the labels will extend exponentially even if we only consider one possible fuel station between them. For instance, if a vehicle visits 35 stops a day, the search space may have as many as \(2^{35}\) labels. In order to resolve this issue, we develop a MIP model to deal with the fuel stations.

Different from the MIP model in Section 3, the choice of fuel stations for each route is independent now and each route can be solved separately. We take route \(r^k\) for instance. First, delete all of the existing fuel stations in \(r^k\) and leave the nodes with the customer and depot vertices only. Let \(r^{k'}\) be the route after deletion and suppose there are \(n_d\) nodes left. In the heuristic method, for any two successive nodes in \(r^{k'}\), we just select the fuel station with the smallest detour as the only candidate. Let binary decision variable \(z^k_{(i)}\) denote the fueling decision for route \(k\) after the \(i\)th node, which is customer node \(r^k_{(i)}\). Let \(z^{k'}_{(i)}\) be 1 if the corresponding vehicle goes to the fuel station after \(r^{k'}_{(i)}\) and be 0 otherwise. We follow the notation in Section 3. For simplification, let \(z_i\) represent \(z^{k'}_{(i)}\), \(i\) represents customer node \(r^{k'}_{(i)}\), \(t_i\) represents \(t^{k'}_{(i)}\), and \(g_i\)
represents $g_i'$. The MIP is as follows.

$$\min t_{n_d}$$

s.t. $$t_i + \frac{\delta_{i,i+1}^+}{\gamma_1} \leq t_{i+1}, \quad \forall i = 1, \ldots, n_d - 1$$

$$t_i + \frac{\delta_{i,i+1}^+ + \delta_{i,i+1}^-}{\gamma_1} + R_f + \frac{G - g_i + \gamma_2^{-1}\delta_{i,i+1}^+}{\gamma_3} + M(z_i - 1) \leq t_{i+1}, \quad \forall i = 1, \ldots, n_d - 1$$

$$0 \leq t_i \leq T, \quad \forall i = 1, \ldots, n_d$$

$$g_i - \frac{\delta_{i,i+1}}{\gamma_2} + Mz_i \geq g_{i+1}, \quad \forall i = 1, \ldots, n_d - 1$$

$$M(1 - z_i) + G - \frac{\delta_{i,i+1}^-}{\gamma_2} \geq g_{i+1}, \quad \forall i = 1, \ldots, n_d - 1$$

$$g_i \geq \delta_{i,i+1}^+ z_i, \quad \forall i = 1, \ldots, n_d - 1$$

$$g_1 = \tau$$

$$g_{d_n} \geq \min_{j \in F} \left( \frac{\delta_{n,j}}{\gamma_2} \right)$$

$$z_i \in \{0, 1\} \quad \forall i = 1, \ldots, n_d$$

The objective function is to minimize the total working time. Constraints (54) and (55) keep track of the working time from node to node. If the vehicle does not go to the fuel station, the working time increases by the traveling time $\frac{\delta_{i,i+1}^+}{\gamma_1}$, which is presented in constraint (54). Otherwise the working time increases by the traveling time, fixed and variable fueling time, which is presented in constraint (55). We use $R_f$ to denote the corresponding fuel station’s fixed fueling time. Constraint (56) gives the boundary of working time. Constraints (57) to (61) are the same as constraints (9) to (13), which ensure the route meets the fuel tank capacity constraint.

In practice, we can easily construct an initial feasible solution by checking the original route plan before the deletion of the fuel stations. Let $z_i$ be 1 if the vehicle goes into fuel station in the original route plan and 0 otherwise. Since the original route plan is feasible, the constructed solution must be a feasible solution for the MIP. In our computational experiments, the MIP can be solved quickly for reasonable sized problems (see Section 5). Similar to the fleet size minimization step, we run it every $\Phi_3$ ALNS iterations. In our experiments, we set $\Phi_3$ to be 100.
4.2.5 Acceptance and Stopping Criteria

Similar to the rule used by [Ropke and Pisinger (2006)], in every ALNS iteration we accept the new solution if it is better than the previous one, or with a probability calculated by equation (49) if it is worse. However, the parameters $T_a$ and $α_a$ are different from $T_s$ and $α_s$. In fact, these two probabilities are updated independently.

The ALNS stops if the improvement of every $φ$ iterations is less than a $β$ percentage, or the total number of iterations reaches $Φ$. We set $T_a = 1000$, $α_a = 0.999$, $φ = 100$, $β = 0.1$ and $Φ = 5,000$.

5 Computational Results

In this section we present the computational results of small, moderate, and large instances. We skip the details about hyper parameter tuning in the ALNS algorithm and because most of the parameters are learnt from [Ropke and Pisinger (2006)] and adjusted based on our data set. We focus more on our unique part instead of the parameter tuning which has been discussed in other papers.

CNG trucks are used as the representation of the AFVs. Data about CNG trucks are collected to help us set the parameters in our model. First we run the proposed ALNS on small instances. These small instances are also solved by the MIP formulation using a standard commercial optimization solver CPLEX. The small instances are used to show the efficiency of the preprocessing and valid inequalities presented in Section 3.2.2 and our adaptive branch-and-cut algorithm in Section 3.2.3 in reducing the computation time in finding an optimal solution. Results also serve as benchmark cases to show that our ALNS can get high quality solutions in a short amount of computation time. Also we use our ALNS to solve moderate and large size instances to explore some interesting insights. The moderate and large size instances include modified cases from [Augerat et al. (1995)] and data from freight demand originating from the Ports of Los Angeles and Long Beach in the Southern California area.

All experiments are performed on a PC with Intel Core i7-4790K CPU and 32 GB RAM, running Linux distribution Ubuntu 14.04 and CPLEX 12.5.

5.1 Results on Small Size Problems

We first create small instances by randomly generating 8 to 10 customers on a 100*100 map. A depot and 3 or 4 CNG fuel stations are also randomly picked within the map. The demands of customers are generated from a uniform distribution between 0 to 0.6 unit. We solve these small instances with CPLEX 12.5, as well
as our proposed ALNS. In order to show the effectiveness of our valid inequalities and adaptive branch-and-cut algorithm, three kinds of formulations, basic, improved and adaptive branch-and-cut, are tested. The basic formulation includes the base MIP formulation and the big $M$ selection. The improved formulation includes everything presented in Section 3.2.2. The adaptive branch-and-cut cases include the improved formulation and the adaptive cut generation method. Note we stop each run after 2 CPU hours if the exact solution procedure has not yet found the optimal solution. The experiments are conducted based on the background of CNG trucks and the parameters are modified from related materials (Laughlin and Burnham, 2016; U.S. Department of Energy, 2014). We set $\gamma_1$, $\gamma_2$ and $\gamma_3$ to be 35 miles/hour, 3.75 miles/GGE (Gallon of Gasoline Equivalent) and 3 GGE/minute. We set $R_f$ (fixed fueling time) to 20 minutes for all fuel stations. Namely every time when a truck goes to a CNG fuel station, its daily working time increases by 20 minutes. The daily working time limitation $T$ and tank capacity $G$ are set to 8 hours (480 minutes) and 53.3 GGE, respectively. The load capacity $L$ for each truck is 1. The initial gas level $\tau_k$ is a random number between $\min_{j \in F} \left( \frac{\delta_j \alpha}{\gamma_2} \right)$ and $G$. 

27
<table>
<thead>
<tr>
<th>Name</th>
<th>CPU Time (s)</th>
<th>Solution Value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Opt Basic</td>
<td>Opt Imp</td>
</tr>
<tr>
<td>8C3F3T1</td>
<td>282.1</td>
<td>56.3</td>
</tr>
<tr>
<td>8C3F3T2</td>
<td>132.0</td>
<td>41.8</td>
</tr>
<tr>
<td>8C3F3T3</td>
<td>150.4</td>
<td>65.1</td>
</tr>
<tr>
<td>8C3F3T4</td>
<td>111.7</td>
<td>27.2</td>
</tr>
<tr>
<td>8C3F3T5</td>
<td>192.3</td>
<td>93.1</td>
</tr>
<tr>
<td><strong>Avg</strong></td>
<td><strong>173.7</strong></td>
<td><strong>56.7</strong></td>
</tr>
<tr>
<td>9C3F3T1</td>
<td>482.7</td>
<td>98.8</td>
</tr>
<tr>
<td>9C3F3T2</td>
<td>3,219.1</td>
<td>692.4</td>
</tr>
<tr>
<td>9C3F3T3</td>
<td>517.4</td>
<td>132.2</td>
</tr>
<tr>
<td>9C3F3T4</td>
<td>270.7</td>
<td>62.1</td>
</tr>
<tr>
<td>9C3F3T5</td>
<td>2,473.2</td>
<td>1,386.2</td>
</tr>
<tr>
<td><strong>Avg</strong></td>
<td><strong>1,392.6</strong></td>
<td><strong>474.3</strong></td>
</tr>
<tr>
<td>10C3F4T1</td>
<td>1,977.4</td>
<td>508.0</td>
</tr>
<tr>
<td>10C3F4T2</td>
<td>2,570.9</td>
<td>2,502.5</td>
</tr>
<tr>
<td>10C3F4T3</td>
<td>2,198.6</td>
<td>768.7</td>
</tr>
<tr>
<td>10C3F4T4</td>
<td>7,200.0</td>
<td>3,649.7</td>
</tr>
<tr>
<td>10C3F4T5</td>
<td>7,200.0</td>
<td>1,769.4</td>
</tr>
<tr>
<td><strong>Avg</strong></td>
<td><strong>4,229.4</strong></td>
<td><strong>1,839.7</strong></td>
</tr>
</tbody>
</table>

Table 1: Results on small instances

Table 1 presents the results for the small instances. In total we conduct experiments with 3 different settings and each setting has 5 random instances. Name 8C3F3T1 means it is the first instance with a setting of 8 customers, 3 CNG fuel stations and 3 trucks. The first part CPU Time has four columns. Column Opt Basic, Opt Imp, Opt B&C are the CPU times (in seconds) used by the basic, improved and adaptive branch-and-cut formulations. Column ALNS is the CPU time (in seconds) for our ALNS. Note that our adaptive branch-and-cut algorithm and ALNS have randomness parts, so we run the adaptive branch-and-cut 5 times and ALNS 10 times, and record the average of the CPU times. The three columns, Opt Basic, Opt Imp and Opt B&C show that our valid inequalities and adaptive branch-and-cut algorithm can save CPU time. On average, our adaptive branch-and-cut algorithm with the improved formulation finds the optimal solution in the shortest amount of time. The second part Solution Value contains the solution values from the MIP.
and the ALNS. The solution values represent the total working time (in minutes) for the fleets in different instances. Column **Opt** is the optimal solution value. For the ALNS, we run 10 times for each instance. Column **# Opt** is the number of times the ALNS obtained the optimal solution. Column **Obj** is the average of the 10 solution values. Column **Gap** is the gap between the average value from the ALNS, which is column **Obj**, and the optimal solution value, which is column **Opt**. The Solution Value part indicates that for each instance, our ALNS can achieve the optimal solution within 10 runs, and the gap between the average value and optimal solution is also very small. In these scenarios, the average gap is below 1.0% and at least 5 out of the 10 runs can identify the optimal solution for each instance.

<table>
<thead>
<tr>
<th>Name</th>
<th>Basic Upper</th>
<th>Basic Lower</th>
<th>Basic Gap</th>
<th>Imp Upper</th>
<th>Imp Lower</th>
<th>Imp Gap</th>
<th>B&amp;C Upper</th>
<th>B&amp;C Lower</th>
<th>B&amp;C Gap</th>
<th>ALNS Best</th>
<th>ALNS Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>15C4F5T1</td>
<td>1.540.7</td>
<td>837.6</td>
<td>45.6%</td>
<td>1.429.4</td>
<td>915.4</td>
<td>36.6%</td>
<td>1.429.4</td>
<td>918.4</td>
<td>35.7%</td>
<td>1.417.9</td>
<td>1.452.8</td>
</tr>
<tr>
<td>15C4F5T2</td>
<td>1.448.6</td>
<td>831.3</td>
<td>46.2%</td>
<td>1.576.0</td>
<td>949.8</td>
<td>39.7%</td>
<td>1.390.4</td>
<td>991.2</td>
<td>28.7%</td>
<td>1.389.0</td>
<td>1.413.5</td>
</tr>
<tr>
<td>15C4F5T3</td>
<td>1.453.6</td>
<td>721.4</td>
<td>50.4%</td>
<td>1.507.9</td>
<td>783.4</td>
<td>48.6%</td>
<td>1.507.9</td>
<td>818.8</td>
<td>45.7%</td>
<td>1.507.9</td>
<td>1.509.1</td>
</tr>
<tr>
<td>15C4F5T4</td>
<td>1.079.8</td>
<td>730.4</td>
<td>32.4%</td>
<td>1.085.0</td>
<td>827.9</td>
<td>23.7%</td>
<td>1.079.8</td>
<td>864.0</td>
<td>20.0%</td>
<td>1.079.8</td>
<td>1.082.7</td>
</tr>
<tr>
<td>15C4F5T5</td>
<td>1.696.4</td>
<td>783.8</td>
<td>53.8%</td>
<td>1.593.3</td>
<td>906.0</td>
<td>43.1%</td>
<td>1.593.3</td>
<td>943.9</td>
<td>40.8%</td>
<td>1.593.3</td>
<td>1.611.9</td>
</tr>
<tr>
<td>Avg</td>
<td>1.443.8</td>
<td>780.9</td>
<td>45.0%</td>
<td>1.458.3</td>
<td>876.5</td>
<td>38.1%</td>
<td>1.400.1</td>
<td>907.3</td>
<td>34.2%</td>
<td>1.397.6</td>
<td>1.414.0</td>
</tr>
</tbody>
</table>

Table 2: Results comparison between different solution methods

We continue to run more instances to show the efficiency of our improved formulation and adaptive branch-and-cut algorithm, as well as the ALNS. Table 2 shows the results of larger instances which can not be optimally solved within 2 hours of CPU time by the basic, improved or branch-and-cut method. That is, after 2 hours of CPU time, we record the lower and upper bounds. As to the ALNS, it converges to a stable solution in a matter of minutes, which is significantly shorter than the other three methods. Thus we do not list its CPU time in the table. In total we run experiments with a setting which has 15 customers, 4 CNG fuel stations and 5 trucks. We run 5 random instances and each instance is solved by three methods which are basic, improved, adaptive branch-and-cut and ALNS. The three group of columns (Basic, Imp, B&C) in Table 2 shows the upper and lower bounds as well as the gaps for these three methods. Note that since the adaptive cut generation has randomness, the columns in the group B&C are the average of 5 replicates. The last group of columns (ALNS) record the best (Best) and average (Avg) solutions within the 10 replicates. Comparing the three exact methods, our adaptive branch-and-cut algorithm always has the lowest upper bound and the gap is also the smallest. The ALNS has a good performance in terms of the solution quality. In some instances, like 15C4F5T1, the best ALNS solution may even be better than the best upper bound of the exact methods.

29
5.2 Results on Moderate Size Problem

In order to explore the effectiveness of the CNG trucks, we conduct experiments on moderate size problems. The moderate instances are modified from the test case A-n33-k5 by Augerat et al. (1995). In the original test case, there are 32 customers and 1 depot randomly allocated on a 100*100 map. Each customer is assigned with a random number representing the demand volume. Each truck has identical capacity limitation and delivers cargo to customers with a tour starting and ending at the depot. The optimal solution for this problem is known in the literature and in the optimal solution, 5 trucks are used to deliver products to the customers. We modify the test case by adding the tank capacity and daily traveling distance constraints to the trucks so that they are more like CNG trucks. The truck speed $\gamma_1$, fuel consumption rate $\gamma_2$, refueling rate $\gamma_3$, and fixed fueling time $R_f$ are set to 35 miles/hour, 3.75 miles/GGE, 3 GGE/minute and 20 minutes, which are identical with Section 5.1. Note that the previous known optimal solution may no longer be feasible with these added constraints. We also randomly select several points within the map to be CNG fuel stations. We assume that there is no CNG fuel station at the depot and all trucks in the fleet need to use the public CNG fuel stations. If we develop a route plan for a single day, all trucks are likely to return to the depot with nearly empty tanks, which will have an influence on the next day. Thus we treat the customers’ demands as daily, rather than a single time, and extend the schedule to be 5 days with the same repeated daily demand. We require that all trucks need to be back to the depot at the end of each day. The heuristic solution method proposed in Section 4 can be used with a slight modification. For the construction of an initial feasible solution in Section 4.1, an extra requirement of minimal gas level is added at the end of a day so that the truck has enough fuel to go to the nearest CNG fuel station on the next day. Only nodes that belong to the same day can be manipulated by the ALNS steps proposed in Section 4.2 except for the CNG fuel station optimization in Section 4.2.4, which is optimized with the nodes for 5 days, since the fueling decision for a truck is made based on its gas level within the 5 days. For daily working time $T$, tank capacity $G$ and load capacity $L$, we set to 8 hours (480 minutes), 53.3 GGE, and 1, respectively. For initial gas level $\tau_k$, we generate it by using the same method as in Section 5.1.

<table>
<thead>
<tr>
<th></th>
<th>Instance 1</th>
<th>Instance 2</th>
<th>Instance 3</th>
<th>Instance 4</th>
<th>Instance 5</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Time</td>
<td>Fuel</td>
<td>CPU</td>
<td>Time</td>
<td>Fuel</td>
</tr>
<tr>
<td>3</td>
<td>6,742.5</td>
<td>18.7</td>
<td>291.0</td>
<td>6,885.4</td>
<td>19.2</td>
</tr>
<tr>
<td>4</td>
<td>6,676.9</td>
<td>18.4</td>
<td>275.0</td>
<td>6,805.0</td>
<td>19.1</td>
</tr>
<tr>
<td>5</td>
<td>6,642.3</td>
<td>18.3</td>
<td>241.2</td>
<td>6,711.8</td>
<td>18.6</td>
</tr>
<tr>
<td>7</td>
<td>6,553.0</td>
<td>17.6</td>
<td>210.7</td>
<td>6,628.3</td>
<td>17.6</td>
</tr>
<tr>
<td>10</td>
<td>6,464.3</td>
<td>17.0</td>
<td>201.4</td>
<td>6,564.3</td>
<td>17.2</td>
</tr>
</tbody>
</table>

Table 3: Results for different number of CNG fuel stations
Table 3 presents the detailed results on the different number of CNG fuel stations. We run 5 instances and each instance has CNG fuel stations increasing from 3 to 10, by gradually adding extra randomly located CNG fuel stations into the map. The instances along the column in Table 3 differ only in the locations of the CNG fuel stations. Column $|\mathcal{F}|$ in Table 3 shows the cardinality of the CNG fuel stations set $\mathcal{F}$, namely the number of CNG fuel stations. For each instance with a certain number of CNG fuel stations, we run the ALNS 10 times and column Time is the average total working time of these 10 runs. Since we make a route plan for 5 days, the total working time includes the 5 trucks’ working time during the 5 days. Column Fuel is the average number of times the 5 trucks use a CNG fuel station during the 5 days. Note that the values of this column vary from 16.5 to 19.2, namely on average a truck visits CNG fuel stations less than one time per day. Though the frequency is low, the fueling decision does influence the total working time. Column CPU is the average CPU time used to solve the instance. From Table 3 we observe that when increasing the number of CNG fuel stations, the total working time decreases, as we expect. The number of times a truck visits a CNG fuel station also decreases. One potential reason is that trucks have more flexibility on the choice of CNG fuel stations when the number of CNG fuel stations increases and drivers are more likely to refuel when their tanks are almost empty. Another reason for the reduction of visiting times is that the total traveling distance decreases. Figure 6 presents the average total working time of these 5 instances. It has a rapid decrease when the number of CNG fuel stations increases from 3 to 4, which means the number of CNG fuel stations is an important factor when the total number is low, and it also follows the law of diminishing marginal utility. Column CPU is the average CPU time used in seconds. On average it takes only few minutes. We compare these solutions with the optimal solution without considering the daily working time and tank capacity constraints. A solution without these constraints can represent a route plan for a fleet of diesel trucks and is a standard solution to a capacitated VRP problem. The optimal solution for the original instance A-33n-5k can be found in the work from [Augerat et al., 1995]
corresponding total working time is 5,665.7 minutes, if the truck speed is 35 miles/hour. Compared with the optimal solution, the average total working time with 3 CNG fuel stations is 19.8% more and with 10 CNG fuel stations is 14.6% more.

<table>
<thead>
<tr>
<th>G</th>
<th>Instance 1</th>
<th>Instance 2</th>
<th>Instance 3</th>
<th>Instance 4</th>
<th>Instance 5</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Time</td>
<td>Fuel</td>
<td>CPU</td>
<td>Time</td>
<td>Fuel</td>
</tr>
<tr>
<td>46.7</td>
<td>6.796.5</td>
<td>21.2</td>
<td>209.4</td>
<td>6.791.7</td>
<td>21.1</td>
</tr>
<tr>
<td>53.3</td>
<td>6.569.0</td>
<td>17.8</td>
<td>211.5</td>
<td>6.613.1</td>
<td>17.6</td>
</tr>
<tr>
<td>60.0</td>
<td>6.493.4</td>
<td>16.3</td>
<td>206.3</td>
<td>6.488.9</td>
<td>14.8</td>
</tr>
<tr>
<td>66.7</td>
<td>6.393.8</td>
<td>13.5</td>
<td>207.0</td>
<td>6.412.9</td>
<td>12.9</td>
</tr>
<tr>
<td>73.3</td>
<td>6.314.5</td>
<td>11.6</td>
<td>206.6</td>
<td>6.352.5</td>
<td>11.1</td>
</tr>
</tbody>
</table>

Table 4: Results for different fuel tank capacities

Table 4 and Figure 7 present the influence of the tank capacity. Table 4 is quite similar to Table 3. Figure 7 shows the tank capacity has a larger influence on the route plan when the tank capacity is low. When the tank capacity is set to 73.3 GGE, the average total working time of 6,308.2 is close to the total working time in the optimal solution without the refueling process and the tank capacity constraints, which is 5,665.7. Indeed, the average total working time only increased by 11.3%. Considering the fixed and variable fueling time, this gap is small. The small gap indicates that refueling is no longer an important factor for CNG trucks under this specific setting, as well as demonstrating the efficiency of our proposed ALNS.

Next we do a sensitivity analysis on the tank capacity to explore its influence on the route plan. We modify the instances by fixing the number of CNG fuel stations at 7 and change the tank capacity from 46.7 GGE to 73.3 GGE. All the other parameters are identical with the previous group instances. Again we run 5 instances and the only difference between these instances is the locations of the CNG fuel stations. Within each instance we run the ALNS with different tank capacities and for each tank capacity level, we run 10 times to get the average results. Table 4 and Figure 7 present the influence of the tank capacity. Figure 7 shows the tank capacity has a larger influence on the route plan when the tank capacity is low. When the tank capacity is set to 73.3 GGE, the average total working time of 6,308.2 is close to the total working time in the optimal solution without the refueling process and the tank capacity constraints, which is 5,665.7. Indeed, the average total working time only increased by 11.3%. Considering the fixed and variable fueling time, this gap is small. The small gap indicates that refueling is no longer an important factor for CNG trucks under this specific setting, as well as demonstrating the efficiency of our proposed ALNS.

32
5.3 Results for Data from the Ports of Los Angeles and Long Beach

We run experiments on the estimation of daily goods delivery data for the Ports of Los Angeles and Long Beach. The data is from the report "SCAG Regional Travel Demand Model and 2008 Model Validation" (LSA Associates, 2012), which was conducted by the Southern California Association of Governments (SCAG) in 2012. Within this model the Southern California region is divided into 4,109 blocks and each block is associated with a pair of longitude and latitude, which represents its location. There is an estimation of light duty diesel truck flow between any pair of blocks. We make some modifications and transfer the demand model to be a routing problem for CNG trucks. The direct distance between every pair of blocks is calculated based on their longitudes and latitudes and use it as the distance between them. For the locations of CNG fuel stations, we collect their locations from the Alternative Fuels Data Center under the U.S. Department of Energy (U.S. Department of Energy, 2017a) and locate them within one of the 4,109 blocks. In total there are 85 blocks that have publicly accessible CNG fuel stations in the Southern California region.

We select the block where the Ports of Los Angeles and Long Beach are located as the depot for our fleet and use the light duty diesel trucks’ truckload trip rates between the depot and the rest of the 4,108 blocks as the daily demands to meet. Within these 4,108 blocks, only the blocks which can be reached by a fully fueled CNG truck without having to stop for refueling are considered. A block which is far away from the port, or far away from the CNG fuel stations are unreachable in our route plan. For each block, we assume the integral part of the demand is fulfilled by FTL (Full Truckload) and the fractional part is fulfilled by LTL (Less Than Truckload). Overall, there are 28 truckloads of FTL demand and 73.7 truckloads of LTL demand allocated in 3,743 blocks and we only consider the 73.7 truckloads of LTL demand for routing.

![Figure 8: Average total working time with different tank capacities for light duty trucks](image)

Similar to the experimental settings in Section 5.2, we make the route plan for 5 days with the same repeated daily demand for light duty trucks to simulate the plan for the fleet over a week. Parameter $T$ is set to 8 hours and all the other parameters are the same as in Sections 5.1 and 5.2. In total, there are
73.7 truckloads of LTL demand. In this experiment, we allow the fleet to use 77 trucks at most, which is the smallest fleet size we can get from the ALNS method. We run experiments with the tank capacity $G$ increasing from 46.7 GGE to 80.0 GGE, eventually to positive infinity to simulate the situation of diesel trucks where the fueling process is no longer a constraint. For a certain level of tank capacity, we run 5 independent replicates and take the average of the best solution in each replicate as the value in the figure. Figure 8 presents the number of refueling stops and the average total working time (in hours) of the fleet in 5 days when changing the tank capacity from 46.7 GGE to 80.0 GGE. On average, these 77 trucks need to refuel 243.3 times in 5 days, namely each truck refuels more than 3 times, when the tank capacity is only 46.7 GGE. When the tank capacity increases to 80.0 GGE, on average, these 77 trucks need to refuel 106.3 times in 5 days, namely each truck only needs to refuel less than twice. Similar to the results in Section 5.2, the tank capacity has a larger influence on the total working time when it is low. When the tank capacity increases from 46.7 GGE to 53.3 GGE, the total working time decreases by 67.3 hours, in which the savings from the fixed fueling time, 16.2 hours, make up only 24.1% of the total saving. In fact, the decrease is mainly from the improvement of routing due to the flexibility from a larger tank capacity. In our case, the benefits of flexibility still exist even when the tank capacity is large. The average daily total working time is 1,396.4 when the tank capacity is as large as 80 GGE. This number is quite close to the one without tank capacity limitation, which is 1,351.6, with only a difference of 3.3%. The program finishes between 90 and 150 CPU minutes in each run, which is fast considering the size of the problem.

![Figure 9: Average total working time with different sizes of light duty truck fleets](image)

Another interesting thing to explore is the effect of the fleet size. In fact, the objective of the conventional VRP with shortest traveling distance implies the minimum number of vehicle, if there are no extra constraints such as tank capacity limitation. We use the same setting as the previous group of experiments and fix the tank capacity at 53.3 GGE. Figure 9 shows the effect of the fleet size on the number of refueling stops and the average total working time (in hours). As the figure shows, the average total working time decreases
when the fleet size increases. However, this reduction stops after the fleet size reaches 87.

6 Conclusions

In this paper, we introduce the VRPAFVFFT to model decisions to be made with regards to the vehicle routes including the choice of CNG fuel stations. This model specially handles several key unique features for CNG vehicles, such as fixed fueling time, sparse allocation of fuel stations and small fuel tank capacity.

We develop some preprocessing steps and valid inequalities to reduce the computing time. Also we propose an adaptive branch-and-cut algorithm to further improve the speed. Experiments with CPLEX show our MIP model works well for this problem and the preprocessing steps, valid inequalities, and adaptive branch-and-cut algorithm can save CPU time. In order to solve a large instance of realistic size, we propose a hybrid heuristic method combining an ALNS with a local search and MIP model. We run experiments on both small and moderate instances as well as the case from the Ports of Los Angeles and Long Beach, and the results show that our proposed method can solve the VRPAFVFFT problem in a short amount of computation time with high quality solutions.

We also get some interesting insights from our experimental results. Based on the results from large instances, we find increasing the number of CNG fuel stations can reduce the workload of a single station, not only due to the allocation of workload, but also due to the reduction in the total number of times trucks use a CNG fuel station. This extra reduction implies the benefits of more CNG fuel stations may be bigger than we expect. Both moderate examples and case from Ports of Los Angeles and Long Beach show that the tank capacity has an impact on the total traveling distance. The impact is big when the mile range is low and still exists but limited when the mile range is high.

Further work may be in two different directions. From the methodology side, the efficiency of the generated cuts can be further improved. From the modeling side, extra constraints from the real world can be considered, such as the queueing time in the fuel stations.

Acknowledgement

We acknowledge Metrans for its kind support of this research.
References


