S©MR

СИБИРСКИЕ ЭЛЕКТРОННЫЕ
МАТЕМАТИЧЕСКИЕ ИЗВЕСТИЯ
Siberian Electronic Mathematical Reports
http://semr.math.nsc.ru
ISSN 1813-3304

Vol. 20, No. 2, pp. 144–145 (2023)
https://doi.org/10.33048/semi.2023.20.???

УДК 519.8
MSC 90B06

A COLUMN GENERATION HEURISTIC FOR A PETROL STATION REPLENISHMENT PROBLEM WITH COMPLEX UNLOADING RULES

B.G. GAVRISH AND Y.A. KOCHETOV

Communicated by P.P. Petrov

Abstract: The paper addresses a new single-depot multi-trip petrol station replenishment problem originating from a Russian petroleum company. We assume that each underground reservoir at each petrol station can be serviced in only one trip, and the service time of the station depends on the number of hoses on the truck. The truck's compartment closest to its cabin must be emptied last. The goal is to minimize the total size of the heterogeneous truck fleet required to service all the stations. We develop a linear 0-1 formulation for an exact approach and a heuristic based on the column generation method to obtain lower and upper bounds on the optimum. The approach is tested on a semi-synthetic dataset, and the bounds differ by at most two trucks in most cases.

Keywords: vehicle routing, matheuristic, heterogeneous fleet, MIP formulation.

Gavrish, B.G., Kochetov, Y. A. A column generation heuristic for a petrol station replenishment problem with complex unloading rules.

The research is carried out within the framework of the state contract of the Sobolev Institute of Mathematics (project FWNF-2022-0019).

Received January, 1, 2023, Published December, 31, 2023.

1 Introduction

In fuel distribution systems, petrol replenishment plays an important role in reducing expenses. These activities include procurement, transportation of products from depots to stations, and inventory management. Classification, models, and exact algorithms for multi-compartment multi-period delivery problems can be found in [1]. Typically, the objective of these models is to maximize a profit function or minimize the size of the truck fleet. The corresponding mixed-integer linear programs contain numerous binary variables and are computationally challenging. Therefore, researchers often develop various heuristic approaches. As a rule, the set of admissible truck routes is restricted by implementing an iterative procedure that includes more and more routes for each truck by solving a general formulation for each truck separately [2], or by splitting the set of stations into clusters based on proximity and solving each subproblem independently [3].

In this work, we consider a specific case of this problem inspired by the workflow of a Russian petroleum company. In our formulation, we assume that each truck must leave the depot fully loaded, and each of its compartments can be used to fill only one underground reservoir. As a result, we obtain a pure 0-1 formulation where the existence of a feasible solution is non-trivial. To address this challenge, we introduce lower and upper demand bounds for each underground reservoir and allow some reservoirs with a zero lower bound to be ignored. We focus on the single-shift case, but in practice, we deal with a multi-period delivery problem and plan to use the single-shift problem as a subproblem in future work.

We apply the well-known column generation approach to this problem. One of the first papers in which this approach is applied to the petrol replenishment problem is [4]. They employ the branch-and-price method for this purpose. A recent example using the same technique is [5], where the authors apply column generation with a diving subroutine in the branch-and-price method for a real-world multi-depot, multi-period petrol replenishment problem. Other studies have used column generation itself in similar settings as a way to calculate the lower bounds on the optimal objective value. The authors of [6] study a formulation similar to the one in this paper. An Adaptive Large Neighborhood Search heuristic is evaluated on data from a Chinese petroleum transportation company, compared against exact results from a MILP model and lower bounds from a column generation approach. The authors of [7] generate daily schedules for vessels (oil tankers) using column generation to construct integer solutions.

We present a novel formulation of the petrol replenishment problem and propose a 0-1 linear model for both exact and heuristic approaches. Each truck leaves the depot fully loaded and can pour fuel from each compartment into only one reservoir. Trucks may be equipped with multiple hoses that allow simultaneous filling of multiple reservoirs. For example, with two hoses,

we need to partition the set of compartments into two subsets to minimize the station service time, assuming a constant pouring speed.

The compartment closest to the cabin must be emptied last. We allow split deliveries to reduce the truck fleet size: each station can be visited multiple times per shift, but each reservoir must be filled in a single trip. To solve the problem and minimize the fleet size, we design a column generation heuristic based on the exact solution to the restricted master problem. The pricing problem is the most computationally difficult step of the method. We solve it in parallel for each truck and impose limits on the maximum number of stations per trip and the set of neighboring stations. We study the influence of these limits on the upper and lower bounds on the optimum.

Our computational results indicate that limiting trips to two stations is a very restrictive constraint. Allowing four or five stations leads to similar results, while three or four stations provide the best balance between running time and solution quality. The gap between the lower and upper bounds is not greater than two trucks in most cases. Therefore, we conclude that the method can generate reasonable daily plans with most trucks completing two trips per shift.

The paper is structured as follows. In Section 2, we describe the model's assumptions and explain our motivation for multiple trips for servicing a single station. Section 3 presents a 0-1 linear programming formulation. In Section 4, the master and pricing problems are introduced for the column generation approach. Section 5 presents our computational results for a semi-synthetic dataset based on real-life instances. Section 6 concludes the work.

2 Problem statement

Let K be the set of heterogeneous trucks and V be the set of stations, including the depot. Let c_{ij} be the travel time associated with the arc (i, j), $i, j \in V$. Let ϕ be the time to pour 1000 liters of petrol into an underground reservoir, and θ be the time needed for visit documentation per station. Let Δ_k be the time required to completely fill and prepare an empty truck $k \in K$ for a route. Each truck is characterized by the set of compartments W_k, each with capacity Q_{wk} for compartment $w \in W_k$, and by its set of hoses S_k. We assume that the hose 0 is used for the longest pouring time during each station visit. Let v_{ik} be a Boolean parameter indicating whether truck k is compatible with station i. Each station has a set of reservoirs, each containing one of the product lines: $(92, 95, 98, 100, \mathrm{DT})$. For each reservoir, we know its product type p and lower and upper bounds l_p and u_p on the quantity of this product to be delivered, where $0 \leq l_p \leq u_p$. Each truck must complete all of its trips and return to the depot within the time window $[0, H_k]$. The objective is to service all stations using the minimal size truck fleet.

2.1. Assumptions.

(1) There are several types of trucks; the fleet is limited and heterogeneous.
(2) Each truck has between 3 and 6 compartments with known capacities ranging from 5,000 to 16,000 liters. Each truck is loaded at the depot. The quantity of product loaded into each compartment is equal to its capacity.
(3) The product from each compartment is transferred into one underground reservoir only.
(4) Each underground reservoir is filled in a single trip.
(5) Each truck leaving the depot must be completely filled.
(6) The compartment closest to the cabin must be emptied last.
(7) Each truck can simultaneously fill a number of reservoirs, limited by the number of its hoses.
(8) The capacity of each station's underground reservoirs is known (20,000 - 50,000 liters), and each reservoir contains only one product.
(9) The depot has sufficient products for an arbitrary replenishment plan.
(10) Only a single working shift is considered; all station orders must be delivered during this shift.
(11) Several trips can be performed by the same truck as long as they can be done within the allowed working hours per shift.
(12) Travel times are constant. Service time depends on the set of reservoirs being serviced simultaneously by a truck using its hoses.
(13) Constraints regarding truck-station compatibility and station-station travel are known.

2.2. Split delivery motivation. We assume that each station can be visited multiple times per shift, but each reservoir must be filled in only one trip. This assumption broadens the scope of our formulation compared to cases where all station orders must be completed in a single visit (see, for example, [2]). Savings from this approach were demonstrated in [8] for the well-known capacitated vehicle routing problem. Now we illustrate how it can lead to a fleet reduction in the context of the new problem.

Suppose that each truck can perform only a single trip. There are three trucks, each with 5 compartments of 5,000 liters. We also consider three stations with the following orders: stations 1 and 3 each have three reservoirs, and station 2 has four reservoirs. We deliver between 5,000 and 10,000 liters to each reservoir.

We need three trucks to service the stations without split deliveries (one station per truck). Alternatively, the first truck can service all the orders at the first station and the first two orders at the second station, while the second truck fills the remaining reservoirs (see Fig. 1). This basic intuition remains valid in other settings and allows for much more flexible plans. Note that this still corresponds to the 'unsplit-unsplit' case according to the classification in [1].

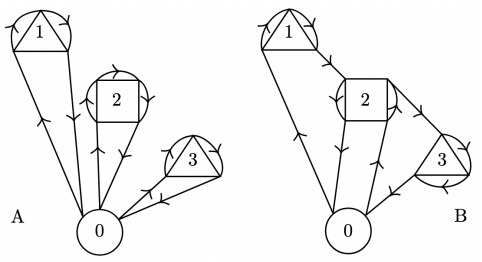

FIG. 1. (A): unsplit deliveries, (B): split deliveries

3 Integer linear programming formulation

For convenience and to simplify the model, we represent each station $f \in V$ by its set of reservoirs R_f and denote the total set of reservoirs, including the depot, as I. The travel time matrix is recalculated with $c_{ij} = 0$ for reservoirs i and j at the same station. As each truck can perform multiple trips, we introduce the set M_k of trips for truck k. We assume that each set M_k is small, with at most M elements.

Now we introduce the decision variables.

- $y_k \in \{0, 1\}$: whether truck k is used
- $g_{km} \in \{0, 1\}$: whether truck k conducts trip m
- $h_{fkm} \in \{0, 1\}$: whether truck k visits station f in trip m
- $z_{ikm} \in \{0, 1\}$: whether truck k fills reservoir i in trip m
- $x_{ijkm} \in \{0, 1\}$: whether truck k travels from i to j in trip m
- $t_{ikwsm} \in \{0, 1\}$: whether truck k uses compartment w to fill reservoir i with hose s in trip m.

According to the model's assumptions, we arrive at the following 0-1 linear formulation.

$$\sum_k y_k \to \min \tag{1}$$

$$t_{ikwsm} \le v_{ik} \qquad \forall (i, k, w, s, m) \tag{2}$$

$$\sum_{i,j,m} c_{ij} x_{ijkm} + \sum_m \Delta_k g_{km} + \frac{\phi}{1000} \sum_{i,w,m} t_{ikw0m} Q_{wk} + \theta \sum_{f,m} h_{fkm} \le H_k \qquad \forall k \tag{3}$$

$$l_i \le \sum_{w,k,m,s} t_{ikwsm} Q_{wk} \le u_i \qquad \forall i > 0 \tag{4}$$

$$g_{km} \le y_k \qquad \forall (k, m) \tag{5}$$

$$\sum_{i,s} t_{ikwsm} = g_{km} \qquad \forall(w,k,m) \quad (6)$$

$$\sum_{j} x_{jikm} \geq \sum_{s} t_{ikwsm} \qquad \forall(i>0,k,m,w) \quad (7)$$

$$t_{ikwsm} \leq z_{ikm} \qquad \forall(i>0,k,w,s,m) \quad (8)$$

$$\sum_{k,m} z_{ikm} \leq 1 \qquad \forall(i>0) \quad (9)$$

$$\sum_{j>0} x_{0jkm} = g_{km} \qquad \forall(k,m) \quad (10)$$

$$\sum_{i} x_{ijkm} = \sum_{i} x_{jikm} \qquad \forall(j,k,m) \quad (11)$$

$$\sum_{i,j\in S} x_{ijkm} \leq |S| - 1 \qquad \forall(k,m,S) : 2 \leq |S| \leq |W_k|, S \subset I \setminus \{0\} \quad (12)$$

$$t_{ik0sm} \leq x_{i0km} \qquad \forall(i,k,s,m) : i > 0 \quad (13)$$

$$\sum_{s} t_{ikwsm} \leq 1 \qquad \forall(i,k,w,m) : i > 0 \quad (14)$$

$$t_{ikwsm} \leq h_{fkm} \qquad \forall(f,i,k,w,m) : i \in R_f \quad (15)$$

$$\sum_{i,w:i\in R_f} t_{ikw0m} Q_{wk} \geq \sum_{i,w:i\in R_f} t_{ikwsm} Q_{wk} \qquad \forall(f,k,s>0,m) \quad (16)$$

In the objective function (1), we minimize the truck fleet size. To do so, we must not service reservoirs with incompatible trucks (2), must respect the time constraint for each truck (3), and must adhere to the lower and upper bounds for each reservoir (4). (5) is a technical condition linking the fleet by variables y_k to the trip variables g_{km}. Constraint (6) ensures that each truck leaves the depot fully loaded. A truck can use its compartments to fill an underground reservoir only if it arrives there (7). Constraints (8) and (9) ensure that each underground reservoir is filled in a single trip.

Additional trip constraints include: (10) ensures that a truck leaves the depot if and only if the trip is conducted. Each truck must leave a reservoir if it visits it (11). Condition (12) is the well-known subtour elimination constraint, ensuring that the subset S of reservoirs does not exceed the number of the truck's compartments. Constraint (13) guarantees that the compartment closest to the cabin ($w = 0$) is emptied last. (14) ensures that one compartment cannot be emptied by two hoses simultaneously. Constraint (15) determines whether truck k visits station f in trip m. Finally, condition (16) ensures that hose 0 is used for the largest amount of fuel at each station.

This is needed for correct calculation of the service time for each trip and each station.

This formulation results in a rapid increase in the number of variables and constraints. To tackle the problem, we design a column generation heuristic based on the exact solution to the restricted master problem. To eliminate non-promising solutions, we impose limits on the maximum number of stations per trip and the set of neighboring stations. As we show below, this can reduce running time, but may increase the required truck fleet size.

4 Column generation approach

We reformulate the problem (1)–(16) as a special case of the set covering problem with a large set R of admissible routes for the trucks. A route $r \in R$ is admissible for truck k if the truck starts from the depot, visits some reservoirs, fills all of them, and returns back to the depot empty. The compartment closest to its cabin is emptied last. For each admissible route r, we can calculate its duration σ_{kr}, which must not exceed the shift length H_k, and a 0-1 indicator b_{ikr} indicating whether the fuel was delivered to reservoir i. Note that the optimal strategy for using the truck's hoses is applied here to compute parameter σ_{kr}. We also introduce new variables $g_{kr} \in \{0, 1\}$: whether truck k performs route r. It is easy to see that our problem is equivalent to the following one:

$$\sum_k y_k \to \min \tag{17}$$

$$\sum_{k,r} b_{ikr} g_{kr} = 1 \qquad \forall i : l_i > 0 \tag{18}$$

$$\sum_{k,r} b_{ikr} g_{kr} \leq 1 \qquad \forall i : l_i = 0 \tag{19}$$

$$g_{kr} \leq y_k \qquad \forall (k, r) \tag{20}$$

$$\sum_r \sigma_{kr} g_{kr} \leq H_k \qquad \forall k \tag{21}$$

Following the column generation framework, we need to find the optimal solution to the linear programming relaxation. To this end, we heuristically generate an initial feasible solution with a small subset of routes and iteratively enlarge it by including the routes with the most negative reduced cost identified through the pricing problem. At the final stage of our procedure, we obtain the lower bound for the optimal value and compute the upper bound as the exact 0-1 solution for the final subset of admissible routes. A similar approach has been successfully applied to many NP-hard combinatorial problems (see, for example, [9], [10], [11]).

4.1. Initial solution. We will use a simple fast heuristic to generate an initial feasible solution. This stage is less critical if a large fleet is available. Otherwise, additional effort is required to verify the problem's feasibility. Assuming the problem has many solutions, we generate a subset of routes randomly, starting from the depot. The probability of each move depends on the distance to the neighboring station. For a sufficiently large set of admissible routes, we solve the master problem (17) – (21) and use the solution as the initial set of routes.

4.2. Route generation. We replace constraint (20) with a compact one

$$My_k \geq \sum_r g_{kr}, \qquad \forall k$$

and consider the dual problem:

$$\sum_{i:l_i>0} \alpha_i - \sum_{i:l_i=0} \alpha_i - \sum_k \delta_k \to \max \tag{22}$$

$$M\beta_k + H_k\gamma_k - \delta_k \leq 1 \qquad \forall k \tag{23}$$

$$\sum_{i:l_i>0} b_{ikr}\alpha_i - \sum_{i:l_i=0} b_{ikr}\alpha_i - \beta_k - \sigma_{kr}\gamma_k \leq 0 \qquad \forall (k,r) \tag{24}$$

$$\alpha_i \in \mathbb{R} \qquad \forall i : l_i > 0 \tag{25}$$

$$\alpha_i \geq 0,\ \beta_k \geq 0,\ \delta_k \geq 0,\ \gamma_k \geq 0 \qquad \forall k,\ \forall i : l_i = 0 \tag{26}$$

For a small subset of routes, we can find the optimal solution $\alpha_i^*,\ \beta_k^*,\ \delta_k^*, \gamma_k^*$ to this linear program. We now need to verify whether constraint (24) holds for all other truck-route pairs. To this end, we create the following pricing problem for each truck with new decision variables:

- $x_{ij} \in \{0,1\}$: whether the truck visits reservoir i directly after reservoir j
- $h_f \in \{0,1\}$: whether station f is visited
- $z_{iws} \in \{0,1\}$: whether compartment w is devoted to filling reservoir i with hose s
- $y_i \in \{0,1\}$: whether reservoir i is filled
- $u_i \geq 0$: auxiliary variables for subtour elimination.

The pricing problem for each truck has the following form:

$$\sum_i \alpha_i^* y_i - \beta_k^* - \\ - \gamma_k^* \Big(\sum_{i\neq j} c_{ij}x_{ij} + \Delta_k + \frac{\phi}{1000}\sum_w z_{iw0}Q_w + \theta\sum_f h_f\Big) \to \max \tag{27}$$

$$z_{iws} \leq v_{ki} \qquad \forall (i>0, w, s) \tag{28}$$

$$\sum_{i\neq j} x_{ij} = \sum_{i\neq j} x_{ji} \qquad \forall j \quad (29)$$

$$\sum_{j>0} x_{0j} = 1 \qquad (30)$$

$$z_{jws} \leq \sum_{i\neq j} x_{ij} \qquad \forall (j>0, w, s) \quad (31)$$

$$\sum_{i\neq j} x_{ij} \leq \sum_{w,s} z_{jws} \qquad \forall j>0 \quad (32)$$

$$u_i - u_j + |W_k| x_{ij} \leq |W_k| - 1 \qquad \forall (i>0, j>0) \quad (33)$$

$$l_i y_i \leq \sum_{w,s} z_{iws} Q_w \leq u_i y_i \qquad \forall i>0 \quad (34)$$

$$z_{iws} \leq y_i \qquad \forall (i>0, w, s) \quad (35)$$

$$\sum_{i,s} z_{iws} = 1 \qquad \forall w \quad (36)$$

$$z_{iws} \leq h_f \qquad \forall (f, i, w, s) : i \in R_f \quad (37)$$

$$\sum_{i\neq j} c_{ij} x_{ij} + \Delta_k + \frac{\phi}{1000} \sum_{i,w} z_{iw0} Q_w + \theta \sum_f h_f \leq H_k \qquad (38)$$

$$\sum_{i,w:i\in R_f} z_{iw0} Q_w \geq \sum_{i,w:i\in R_f} z_{iws} Q_w \qquad \forall (f, s>0) \quad (39)$$

$$z_{i0s} \leq x_{i0} \qquad \forall (i>0, s) \quad (40)$$

If the optimal value is positive, we enlarge our subset of admissible routes by including the optimal solution of the pricing problem and repeat the process. Otherwise, we have obtained a lower bound for the minimal fleet size. Typically, this lower bound is extremely tight [12], but the computational time can be significant. The most important question here is the following. How many iterations are needed to obtain the lower bound, and what can be achieved before the stopping criterion is met, especially for large-scale instances with long computation times?

4.3. Pricing problem resolution. It is worth noting that, even for instances of moderate size (e.g., 200 reservoirs or more), the pricing problem becomes computationally hard to be solved exactly, even when using parallel processing with commercial solvers. Iterations of the pricing problem may take up to a few hours. To address this issue, we propose an alternative approach for cases with a small number of stations per route. In real-life applications, the set of available routes between stations is constrained by the company operating the truck fleet. This reduces the service time. According to company rules, a truck can only move to one of the few nearest stations. This implies that the set of feasible routes for each truck is relatively small and can be precomputed for daily use. In other words, this procedure can be considered a preprocessing stage.

To generate all admissible routes, we use a modified Depth-First Search algorithm to traverse the graph of reservoirs, combined with an optimized Tank Truck Loading Procedure (TTLP) to check the feasibility of each route. The graph traversal is conducted in the following way. We select a truck and start from the depot. At each step, we iterate through all feasible next reservoirs, checking that the extended set of reservoirs can be serviced by the given truck. The proposed TTLP procedure returns a second Boolean value indicating whether no reservoir can be added to the current set so that the truck is able to fill all of them. Consequently, the recursion is stopped in this case.

The TTLP procedure is presented below. First, corner cases are handled. When the number of reservoirs equals the number of truck compartments, the allocation problem reduces to a matching problem. If the number of reservoirs is smaller, the problem is solved using dynamic programming. $\mathrm{DP}_i(S) \leftarrow$ *true* if the subset of reservoirs S can be serviced by truck compartments. We also denote by i', w' the last compartment and the last reservoir on the route, respectively. After the procedure, hose allocation is performed by brute force and time feasibility is checked.

procedure TTLP(Q_w, l_i, u_i) // determines feasibility
  $n \leftarrow |\{i\}|$
  $m \leftarrow |\{w\}|$
  if $n = 0$ **then**
    return (false, true)
  else if $n = 1$ **then**
    if $\sum_w Q_w < l_1$ **then**
      return (false, false)
    else if $\sum_w Q_w \le u_1$ **then**
      return (true, true)
    else return (false, true)
  else if $n > m$ **then return** (false, false)
  else if $n = m$ **then**
    $M \leftarrow \text{match}(\{i\}, \{w\} \mid (i', w') \in M)$ // max. matching [13]
    if $|M| = n$ **then return** (true, false)
    else return (false, false)
  else // case $1 < n < m$
    if $Q_{w'} > u_{i'}$ **then return** (false, true)
    $F_i \leftarrow \{\, S \subseteq \{w\} \backslash w' : l_i \le \sum_{w \in S} Q_w \le u_i \,\}$
    $F_{i'} \leftarrow \{\, S \subseteq \{w\} \backslash w' : l_i \le Q_{w'} + \sum_{w \in S} Q_w \le u_i \,\}$
    $\text{DP}_1(S) \leftarrow \mathbf{1}_{F_1}(S)$
    for $i = 2, \ldots, n$ **do** $\text{DP}_i(S) \leftarrow$ false
    for $i = 2, \ldots, n$ **do**
      for $S \subseteq \{w\} \backslash w'$ **do**
        for $S' \in F_i$ **do**
          if $S' \subseteq S$ **then**
            $\text{DP}_i(S) \leftarrow \text{DP}_i(S) \vee \text{DP}_{i-1}(S \setminus S')$
    $\text{ANS} \leftarrow \text{DP}_n(\{w\} \backslash w')$
    return (ANS, true)
end procedure

The maximization in (27) is then solved by exhaustive search. After having generated all possible routes for each truck, the pricing problem becomes fairly straightforward. It is sufficient to iterate through all the routes at each step of the column generation procedure. As shown in the next section, most admissible routes are rejected by the pricing problem, resulting in a small final subset of routes. This results in a small restricted master problem (17) – (21), which can be easily solved using a solver. This provides an upper bound for the fleet size in the original problem. To obtain the optimal fleet size, we would need to solve the restricted master problem for the entire set of admissible routes.

5 Numerical simulations

The formulation that is studied here has direct practical applications. Petrol companies face vehicle routing problems on a daily basis. Several features of real-life problems studied in this work are worth highlighting.

Firstly, most reservoirs are nearly full, so the total number of orders is relatively low, even for a large network of stations (e.g., out of 120 stations with approximately 5 reservoirs each, only 60 reservoirs may require filling). Secondly, the truck-station compatibility constraints arise due to different truck types. There are two key features: the type of drain (left-sided, right-sided, or double) and the truck type (heavy or not). The type of station is determined by its ability to accommodate these types of truck. Finally, the depot is often located at a considerable distance from most stations. As a result, most trucks complete no more than two trips during a 12-hour shift.

To construct problem instances, we make assumptions based on statistical analysis of real-life data.

We set the number of hoses to 2 for each truck, and assume the same working day duration for all trucks, with $\phi = 3$ minutes and $\theta = 15$ minutes.

The sizes of the truck compartment are modeled using a lognormal distribution with parameters $\mu = 9$, $\sigma = 0.3$. A truck has four compartments with probability 0.2, five compartments with probability 0.5, and six compartments with probability 0.3. A truck is considered heavy if its total capacity exceeds 40,000 liters. Forty percent of trucks are two-sided, while 30% are left-sided and 30% are right-sided. We assume that it takes three minutes to load and unload 1,000 liters, so $\Delta_k = \frac{\sum_w Q_w}{1000} \times 3$.

Stations are characterized by their reservoirs. The capacity of each reservoir follows a lognormal distribution with $\mu = 10$ and $\sigma = 0.4$, while the occupancy rate follows a beta distribution with $\alpha = 4.5$ and $\beta = 1.4$. The number of reservoirs for the simulations follows a Poisson distribution with $\mu = 6$. Regarding station types, we suppose that 30% of the stations can accommodate heavy trucks and, independently, 40% are two-sided and left and right-sided stations comprise 30% each.

The following strategy is used to choose locations. We place the depot at point $(0, 0)$ and uniformly distribute the stations within the square of $(100, 100)to(200, 200)$. Distances are calculated in the Manhattan metric. The travel times are assumed to be proportional to distance and normalized, so it takes one hour for a truck to travel from the depot to $(100, 100)$.

5.1. Simulation strategy. Our solution approach narrows the set of allowed routes for each truck by introducing two additional restrictions. Firstly, from each station, the truck can only proceed to a certain number of the nearest stations. Secondly, the total number of stations visited by a single truck during one trip from the depot is also usually limited (for example, in [14], the maximum number is set to 2).

We do not wish to set these restrictions arbitrarily. Instead, we perform a grid search to evaluate the performance of different solution strategies on 10 test instances drawn from the same distribution described above with $N = 30$ stations, $K = 15$ trucks. We also impose a time limit of 30 minutes for each run to ensure practical applicability.

5.2. Simulation results. The proposed algorithm was implemented in C++. The MILP problems were solved using the Google OR-Tools library with the CP-SAT solver backend. The experiments were conducted on an Apple M2 Max chip. In the following tables, R_1 denotes the maximum number of stations visited by a truck per trip, and R_2 represents the number of available neighboring stations from each station.

$R_2 \setminus R_1$	2	3	4	5
12	9 / 10	6 / 10	1 / 10	0 / 10
8	7 / 10	9 / 10	5 / 10	3 / 10
4	4 / 10	9 / 10	10 / 10	9 / 10

TABLE 1. Number of successfully completed runs

The results presented in Table 1 demonstrate that excessive constraints can lead to problem infeasibility, as evidenced by only 4 successful runs in the most constrained scenario. Conversely, relaxing these constraints results in increased computational complexity and longer solution times. From a practical implementation perspective, moderately constrained parameters (such as $R_1 = 3$, $R_2 = 4$) appear to offer the optimal balance between feasibility and computational efficiency. Restricting routes to 3 stations and 4 neighboring options reduces computational complexity while maintaining solution quality, as overly strict limits ($R_1 = 2$) lead to infeasibility, and relaxed limits ($R_1 = 5$) increase runtime without significant fleet reduction.

$R_2 \setminus R_1$	2	3	4	5
12	1.16 (0.78)	13.13 (6.10)	12.17 (0.00)	–
8	0.64 (0.42)	6.10 (3.93)	14.14 (6.63)	18.17 (8.64)
4	0.20 (0.17)	1.40 (1.38)	4.23 (3.73)	4.24 (3.27)

TABLE 2. Route preprocessing time average and its standard deviation (minutes)

Table 2 provides deeper insight into the computational complexity under varying constraints by examining preprocessing times (the duration required

to generate all feasible routes). The results show significant variation in the processing times for different combinations of parameters, with more relaxed constraints generally requiring longer preprocessing periods.

$R_2 \setminus R_1$	2	3	4	5
12	6.22 (0.74)	5.00 (0.33)	4.00 (0.00)	–
8	6.43 (0.65)	5.33 (0.67)	5.00 (0.40)	4.67 (0.44)
4	6.75 (0.38)	5.56 (0.72)	5.30 (0.82)	5.22 (0.35)

TABLE 3. Average number of trucks used and its standard deviation

A notable finding revealed in Table 3 is the substantial reduction in the number of trucks required when the maximum number of stations visited per trip is increased from 2 to 3. This difference suggests that limiting routes to only 2 stations per trip is overly restrictive and may lead to suboptimal resource utilization in the current formulation.

Further analysis focuses on the configurations that demonstrated the best performance (achieving at least 9 out of 10 successful solutions as shown in Table 1). Table 4 presents detailed information about the number of columns utilized in each case. While the total number of generated routes varies considerably across instances, the column generation algorithm consistently achieves convergence after adding a relatively small number of additional routes. It is important to mention that the exact solution to the linear problem does not serve as an effective initial solution for the binary integer programming problem, thus validating the choice of a column generation approach for route selection.

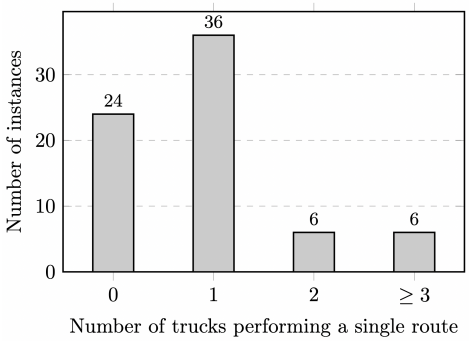

FIG. 2. Distribution of instances based on the number of trucks with a single route

Instance	$R_1 = 3\,/\,R_2 = 4$	4/4	5/4	3/8	2/12
1	115357 10 146	349088 9 159	524621 10 169	490099 9 165	66343 10 68
2	24252 9 124	75924 10 116	128158 9 129	119096 9 154	12805 12 67
3	113074 11 154	541929 11 92	924221 12 74	520067 11 202	29945 12 59
4	20764 8 78	61589 8 146	66399 8 160	84660 8 76	5839 10 35
5	168129 10 112	596907 11 120	877185 11 128	731613 13 93	43629 11 42
6	14437 12 50	143420 9 118	484035 10 173	414452 11 91	27445 11 63
7	330906 14 137	1342994 13 110	–	–	98280 12 96
8	77515 10 136	266839 10 150	401444 10 106	261738 9 139	21360 11 61
9	–	204061 9 152	296980 9 177	162125 9 62	–
10	80379 11 103	298804 9 136	875230 9 111	298362 9 73	19563 12 39

TABLE 4. Number of columns across instances. [1]

[1] In each cell, top–to–bottom: total generated routes, initial solution routes, added routes via CG.

A crucial practical consideration is the distribution of routes among trucks during their 12-hour shifts. The solution results indicate that the majority of trucks complete two routes, efficiently utilizing their designated time. Only a small number of vehicles, as visualized in Fig. 2, are assigned to single routes

per shift, demonstrating that the solution algorithm successfully handles complex unloading constraints while producing efficient schedules.

6 Conclusion

We studied a new formulation of the petrol replenishment problem for a single shift. Based on the specific characteristics of the petrol replenishment in Russia, we introduced a set of novel assumptions that significantly affected the nature of the problem. We presented a new 0-1 linear formulation for the problem, which includes complex unloading rules, and developed a column generation heuristic. Numerical simulations on a semisynthetic dataset demonstrate that the proposed procedure can produce near-optimal solutions.

In future work, it would be of interest to explore generalizations of the problem discussed here. Extending the planning horizon to a week would make the setting more realistic, as current orders are interconnected with past and future ones. In addition, travel times may differ between night and day shifts. Consequently, some stations may need to be visited only during night shifts. Determining the optimal partition of stations between night and day shifts is also a promising research direction.

References

[1] L.C. Coelho, G. Laporte, *Classification, models and exact algorithms for multi-compartment delivery problems*, European Journal of Operational Research, **242**:3 (2015), 854–864.

[2] F. Cornillier, F. Boctor, J. Renaud, *Heuristics for the multi-depot petrol station replenishment problem with time windows*, European Journal of Operational Research, **220**:2 (2012), 361–369.

[3] F. Cornillier, G. Laporte, F.F. Boctor, J. Renaud, *The petrol station replenishment problem with time windows*, Computers & Operations Research, **36**:3 (2009), 919–935.

[4] P. Avella, M. Boccia, A. Sforza, *Solving a fuel delivery problem by heuristic and exact approaches*, European Journal of Operational Research, **152**:1 (2004), 170–179.

[5] A. Bani, I. El Hallaoui, A.I. Corréa, A. Tahir, *Solving a real-world multi-depot multi-period petrol replenishment problem with complex loading constraints*, European Journal of Operational Research, **311**:1 (2023), 154–172.

[6] L. Wang, J. Kinable, T. van Woensel, *The fuel replenishment problem: a split-delivery multi-compartment vehicle routing problem with multiple trips*, Computers & Operations Research, **118** (2020), 104904.

[7] S.M. Al-Yakoob, H.D. Sherali, *A column generation approach for determining optimal fleet mix, schedules, and transshipment facility locations for a vessel transportation problem*, Applied Mathematical Modelling, **37**:4 (2013), 2374–2387.

[8] M. Dror, P. Trudeau, *Split delivery routing*, Naval Research Logistics, **37**:3 (1990), 383–402.

[9] Yu.A. Kochetov, A.V. Ratushnyi, *Upper and lower bounds for the optimum in a temporal bin packing problem*, Trudy Instituta Matematiki i Mekhaniki UrO RAN, **30**:1 (2023), 109–127.

[10] Y. Kochetov, A. Kondakov, *A hybrid VNS matheuristic for a bin packing problem with a color constraint*, YUJOR, **31**:3 (2021), 285–298.

[11] P. Avella, M. Boccia, S. Salerno, I. Vasilyev, *An aggregation heuristic for large scale p-median problem*, Computers & Operations Research, **39**:7 (2012), 1625–1632.
[12] V.M. Kartak, A.V. Ripatti, G. Scheithauer, S. Kurz, *Minimal proper non-IRUP instances of the one-dimensional cutting stock problem*, Discrete Applied Mathematics, **187** (2015), 120–129.
[13] H.W. Kuhn, *The Hungarian method for the assignment problem*, Naval Research Logistics Quarterly, **2**:1–2 (1955), 83–97.
[14] F. Cornillier, F.F. Boctor, G. Laporte, J. Renaud, *An exact algorithm for the petrol station replenishment problem*, Journal of the Operational Research Society, **59**:5 (2008), 607–615.

Boris Georgievich Gavrish
Moscow State University,
Leninskiye Gory, 1-46
119991, Moscow, Russia
Email address: gavrish.boris@list.ru

Yury Andreevich Kochetov
Sobolev Institute of Mathematics,
pr. Koptyuga, 4,
630090, Novosibirsk, Russia
Email address: jkochet@math.nsc.ru