The fleet composition and allocation problem: Model, heuristic and case study

Maria Eduarda Jes*, Alexandre Checoli Choueiri*

Department of Industrial Engineering
Federal University of Paraná, Brazil
E-mail: maria.jes@ufpr.br
E-mail: alexandrechecoli@ufpr.br
*Corresponding author

Abstract: The logistic decision of leasing and allocating a fleet of vehicles to perform deliveries can be burdensome, specially considering the many variables involved in the process. This paper presents the FCAP (fleet composition and allocation problem), the problem of determine which contracts to lease vehicles will be implemented, and after that the allocation of the vehicle to working days considering a planning horizon. This problem arise from a national wide food delivery company from Brazil. We propose a mathematical programming method as well as a LNS heuristic to solve the problem. As a company restriction, we use only open-source solvers to perform the computational tests. Results are promising, leading to a hybrid methodology that used the solver or the LSN heuristic depending on the instance complexity. All tested instances were real problems from the company.

Keywords: Fleet composition, fleet allocation,integer programming

1 Introduction

Road transportation plays a crucial role in the displacement of cargo of all countries. It is the more commonly used and reliable way of transporting goods and services inland (Pasha et al., 2016). The costs involved in using road transportation, however, may account for a big slice of a company profit: it is commonly assumed that transportation costs can get as high as 20% of a product total production cost (Hoff et al., 2010).

Several factors may influence a company on the decision of outsourcing a fleet of vehicles, the complexity of jointly integrate production, inventory and routing may serve as a driven motive toward that end Li et al. (2019). Another simpler reason may be the variety and volumes of goods that are to be transported: the costs of holding and maintaining a personal fleet versus the flexibility of on-demand leasing may not be attractive.

The outsourcing of transportation is put forward trough the commitment of leases, where firms negotiate agreements and their conditions with the fleet providers Ansaripoor and Oliveira (2018). Those contracts encompasses financial aspects of logistics, like total costs, as well as those involved in operational decisions, like the period that a vehicle will be available to perform the deliveries. When those contracts involve large amounts of money,

Copyright © 20xx Inderscience Enterprises Ltd.

the contractor may even define the routing decisions that are to be made by the outsourced fleet. Since a contract for a vehicle is made based on the number of days that it will be available, and there are different contracts for different periods, there is a two dimensional decision that is to be made: how many vehicles of each type of contract is to be leased, and, on what days are they allocated to work on deliveries.

A leased vehicle has a fixed cost of use, and it will be available for k days on a P planning horizon (usually $P > k$). If a vehicle is contracted for k days and was actually used on less than k, the total cost of the contract remains the same. The determination of which days of P the vehicle will be used is hereby called the *allocation problem*. Besides the allocation, the number of vehicles of each contract that are to be used also needs to be determined - this is called the *fleet-composition* problem. The combination of the two decisions is the *fleet-composition and allocation problem* (FCAP).

In this paper we study the FCAP arising from a big food company in Brazil (almost 100 thousand employees worldwide). The company produces various types of processed foods, ranging from oil and butter to cheese, ham and turkey. Although the company possesses a personal fleet of vehicles, with such a variety and volume of products, it becomes too expensive to maintain a fleet composition capable of delivering all of the client demands. Therefore, an extra outsourced fleet needs to be contracted in order to fulfill freight needs.

The current decision process of the company involves two-phases: the first one takes as input client demands on the planning horizon P along with several traffic restrictions, and determines the route that for each vehicle of the fleet. This generates a new demand vector: the number of vehicles that are needed for each day of P. This new input triggers the process of (manually) determining the contracts to use the leased fleet of vehicles, in order to meet this new demand (the FCAP).

In this paper we developed an integer programming (IP) model for the FCAP, as well as a LNS metaheuristic to deal with larger instances. Computational experiments were run in order to evaluate the feasibility of using exact methods (trough the use of solvers) to cope the company problem. The experiments were performed using an open-source solver (SCIP), since the company has no license for a commercial one.

The main contributions of this paper are then: the modeling of a real-case leasing logistic problem and the methodology to solve it, using open-source solvers and metaheuristics.

The paper is organized as follows: in Section 2 the problem is described, and how it is positioned considering the literature. Section 3 presents the mathematical programming model and an algorithm to extract upper-bounds from the parameters. In Section 4 the initial solution and the LNS heuristics are presented, followed by the computational results in Section 5. Finally Section 5 entails some concluding remarks.

2 Problem description and literature positioning

As previously mentioned, the FCAP arose from a Brazilian food company, which has its own personal fleet of vehicles, but also uses third party leased contracts in order to fulfill remaining demands. Figure 1 shows a representation of the current decision process of the company, which helps in understanding the FCAP.

First, all client demands for a given planning horizon are used as an input by the logistics department, in order to compute (via an internal optimization software) the routes and the number of vehicles needed to deliver all requests. It is worth noting that at this level, all vehicles have the same capacity. The explicit output of this computation is of course the

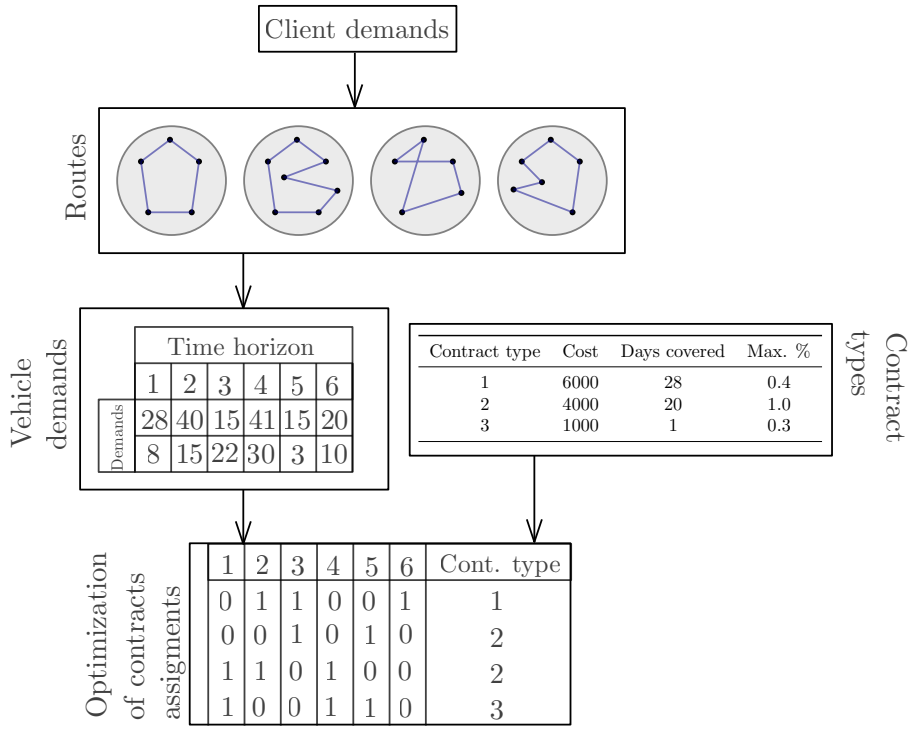

Figure 1 Problem description

routes, but from that it is easy to derive a new demand vector, formed by vehicle demands at each period of time. Although the vehicles have equal capacities, there are different types of lease contracts.

Each lease contract can be defined by a fixed cost, the number of days that the vehicle can be used (Days covered in Figure 1) and the maximum proportion ratio of the vehicle over all the leased fleet. This is a company determination, and takes into the difficulty of hiring fleet for some specific regions. Both the vehicle demands and the contract types are now used as input in the optimization problem (FCAP), where the number of vehicles for each contract type need to be determined, as well as the assignment of the vehicles to days os the horizon. The number of days assigned for each vehicle must not exceed its covered days, the total vehicles of each type should not exceed its max proportion ratio and the vehicle demands for each day must be satisfied, while minimizing the sum of fixed costs.

Figure 1 describes the process for one type of vehicle (capacity) and one region. This procedure is repeated for a number of different regions and vehicle types, generating a huge amount of FCAPs to be solved. Currently the company have no optimization procedure whatsoever to solve the problem, relying only on spreadsheets and managers experience. The process of solving the FCAP takes several weeks, and much rework needs to be done in order to satisfy all of the constraints.

Although the FCAP encompasses multiple common decisions on fright transportation, to the best of the authors knowledge there are no studies directly dealing with it. There are, however, a set of correlated problems that can be of use in the understanding of the FCAP, those papers are briefly described in this section.

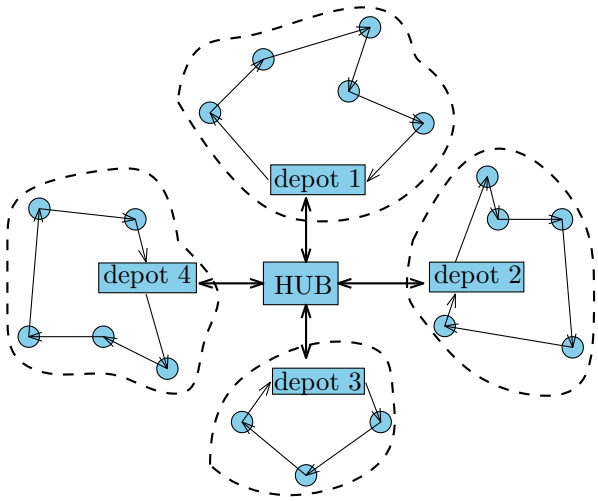

Figure 2 Transportation network of a hub-depot system

As a first correlated problem to the FCAP there is the fleet size and mix vehicle routing problem (FSMVRP). The FSMVRP was first presented in Golden et al. (1984), where the authors determine not only the routes that are to be made by vehicles with the classical VRP constraints, but also the quantities of each type of vehicle, given that different vehicles have different capacities, fixed and variable costs. In the work of Efthymiadis et al. (2023), the authors solve the FSMVRP with vehicle multi-compartment constraints by means of a mathematical model. The input data is pre processed in order to decompose the original problem into multiple sub-problems, which can be then handled by the CPLEX optimization solver. In recent years, the concept of green logistics has gained a lot of attention, due to its practical importance Utama et al. (2020). In this regard, Dönmez et al. (2022) proposes a new FSMVRP variant, where the vehicles are electric and there is the partial recharging by multiple sources of chargers consideration. Th authors develop the MILP model for the problem as well as an Adaptive Large Neighborhood Search (ALNS) metaheuristic to deal with larger instances.

In the work of Zäpfel and Bögl (2012), a problem very similar to the FSMVRP is proposed (vehicle selection problem - VSP), but the authors do not determine the routes of every vehicle, and there is a possibility of leasing the fleet. The problem is based on line haul transport, which comprises the transportation of goods from each local depot of a service area to the central hub and vice versa. Figure 2 depicts the process:

The fleet mix of vehicles is to be determined based on the hub-depot demands, by personal fleet or leasing contracts. The authors propose an IP model for the problem and show that it can be viewed a variant of a multiple generalized assignment problem. In this sense, it will be clear on the next sections that the FCAP is also a variant of a generalized assignment problem (GAP).

Due to the wide range of real life problems that can be formulated as a GAP, the search for effective algorithms to solve it is also high. In the recent work of We Li and Ou (2024), the authors solve the parallel machine scheduling problem considering green constraints, and also show that the problem falls into a GAP variant. To the original GAP we refer to Martello and Toth (1990).

We conclude this section then, positioning the FCAP among similar, but not quite the same, problems: the FSMVRP, the VSP and the GAP.

3 Mathematical model

In this section we describe the IP formulation that models the FCAP, as well as an algorithm to derive upper-bounds on the number of variables.

Considering the following notation:

Sets:

- K: Set of different contracts
- T: set of days

Parameters:

- D_t: truck demand on period $t, t \in T$
- C_k: coverage days of contract of type $k, k \in K$
- U_k: number of vehicles available for the k type of contract type $k \in K$
- P_k: maximum permitted percentage of vehicle from contract type $k, k \in K$
- F_k: fixed cost for using vehicles from contract type $k, k \in K$

Variables:

- x_{kmt}: binary variable indicating if the mth car of contract type k is assigned to be used on day t $k \in K, m \in \{1, ..., U_k\}, t \in T$
- y_{km}: binary variable indicating if the mth car of contract type k is used

The proposed formulation for the FCAP read as follows:

$$\min \sum_{k \in K} \sum_{m \in M} y_{km} F_k \tag{1}$$

Constraints

$$\sum_{t \in T} x_{kmt} \leq y_{km} C_k \qquad k \in K, m \in \{1, ..U_k\} \tag{2}$$

$$\sum_{k \in K} \sum_{m=1}^{U_k} x_{kmt} \geq D_t \qquad t \in T \tag{3}$$

$$\sum_{k \in K} \sum_{m=1}^{U_k} y_{km} = s \tag{4}$$

$$\sum_{m=1}^{U_k} y_{km} \leq s P_k \qquad k \in K \tag{5}$$

$$x_{kmt} \in \{0, 1\} \qquad k \in K, m \in \{1, ..U_k\}, t \in T \tag{6}$$

$$y_{km} \in \{0, 1\} \qquad k \in K, m \in \{1, ..U_k\} \tag{7}$$

It is important to highlight the need of an upper-bound on the number of vehicles (variable x and y) for each type of contract: there has to exist enough vehicles of each type, considering the case where all of the demands are met using exclusively this vehicle. The upper-bounds are held on parameters U_k, and an algorithm to extract them is described right after the model explanation.

The objective function 1 minimizes the fixed cost of all contracts (vehicles) that were used. The first constraint (2) guarantees that a vehicle assigned work days do not exceed what their contract allows(C_k). It also activates the binary variable y_{mk} whenever vehicle m of contract type k is used, thus adding its fixed cost to the objective function.

Constraint number 3 assures that the demand for each day is satisfied, while constraint 4 only sums the number of vehicles of contract k that were used. Finally, constraint 5 provide a limit on the number of vehicles of type k used, considering the maximum allowed ratio (P_k).

As stated before, we need to define upper-bounds on the number of vehicles (for each contract). The idea behind the algorithm is to decompose the demand vector into a sum of unit vectors, and for each of those, use the maximum coverage days of a vehicle to infer the minimum number of vehicles of that type that are needed to attend the demand.

For example, considering a demand vector on a 4 days horizon as $D = [3, 2, 1, 3]$, we can decompose it into the following unit vectors:

$$\begin{aligned} v_1 &= [1, 1, 1, 1] \\ v_2 &= [1, 1, 0, 1] \\ v_3 &= [1, 0, 0, 1] \end{aligned}$$

Since the vehicle assignment on a day is represented as a binary variable on that day, we can now see the demand vector on a more suited perspective. If we consider a vehicle that has a maximum of 2 days of coverage (C on the model), we need 2 of those to attend the unit demand vector v_1. To attend v_2 note that it is also needed 2 vehicles, and one of them will have a day not assigned. Finally, the last vector v_3 needs only one vehicle. The upper-bound is then calculated as the sum of needed vehicles for each unit vector, in this case 5.

Algorithm 1 defines a procedure to decompose a demand vector (D) into a list of unit vectors.

Lines 1 and 2 initializes the list of vectors (L, initially empty) and a variable n, which will hold the number of non-zero elements on a vector. In line 3 the variable min is filled with the minimum demand value of D that is different from zero. Line 4 initializes the first unit vector; for each non zero value in D a 1 value is assigned in v_unit, and 0 otherwise. Following, the for loop in line 6 adds n unit vectors on L.

After that, the demand vector is updated on line 8, where min is subtracted for each value of D that is not 0. The process repeats until n is zero.

With the list L populated with unit vectors, the upper bound for vehicle k (U_k) is easily derived, considering the coverage days C_k (using L_i as the i-eth unit vector from L):

$$U_k = \sum_{i=1}^{|L|} \left\lceil \frac{\sum L_i}{C_k} \right\rceil \tag{8}$$

Algorithm 1: $create\_unit\_vectors$

Data: Demand vector D,coverage days of contract k, C_k
Result: Upper bound of contract k, U_k

```
L ← ∅;
n ← 0 ;
min ← find_min_non_zero(D);
v_unit ← create_unit_vector(D, n);
while n > 0 do
    for i = 1 to min do
        L.add(v_unit) ;
    remove_min(D, min);
    min ← find_min_non_zero(D);
    v_unit ← create_unit_vector(D, n);
```

4 Metaheuristic procedures - constructive and LNS

Despite the simplicity of the model, it can become intractable as the demand grows in volume; that is due to the fact that U_k is directly derived from D, and hence, the number of variables in the model also grows with D.

Single solution metaheuristics are based on the concept of neighborhood. The *neighborhood* of a solution x is a mapping $N(x)$ leading to a set of solutions. Neighborhoods N are usually implemented with *move operators*; they apply changes in the solution x leading to a new set of solutions.

There are two decision levels on the FCAP: i-) which type of contract to select, and ii-) which days assign for the vehicle to work. Considering only the second level of decision, on a horizon with $|D|$ days and a vehicle k with C_k days for coverage, the number of different possible assignments is given by the simple combination $\binom{|D|}{C_k}$. If we consider the planning horizon for a month, and a contract type that allows for 20 days of coverage, the number of possible assignments is 3.0045015 E+7. That alone suggests that an explicit neighborhood definition for the FCAP will become exponentially large to be explored.

An algorithm that searches a neighborhood that grows exponentially with the instance size is considered as belonging to the class of VLSNS (Very Large Scale Neighborhood Search) Ahuja et al. (1998). One of those methods proposed by Shaw (1998) is the *Large Neighborhood Search* - LNS.

The principle of the LNS is very simple, and it fits well to explore large neighborhoods on an implicit manner. This is achieved by defining the neighborhood by a *destroy* and a *repair* operator. The destroy operator deconstruct part of the current solution, while de repair procedure rebuilds the solution again. The neighborhood $N(x)$ of a solution is then defined as the set of solutions that can be achieved by first using the destroy, and then the repair operator. The pseudo-code of an LNS can be seen in Algorithm 2.

The algorithm maintains only 3 variables; x^b as the best solution obtained, x^t is the temporary solution, that can be discarded or promoted to current solution. The $d(\cdot)$ procedure represents the destroy operator, and $r(\cdot)$ the repair.

We first present the greedy randomized procedure designed to construct an initial solution, and then the destroy and repair operators used on the LNS are described (since they rely on similar ideas of the constructive heuristic).

Algorithm 2: LNS

Data: feasible solution x
Result: best solution found x^b

```
x^b ← x;
while criterion not satisfyied do
    x^t ← d(x);
    x^t ← r(x);
    if accept(x^t,x) then
        x ← x^t;
    if c(x^t) < c(x^b) then
        x^b ← x^t;
```

4.1 Greedy randomized constructive heuristic

The data structure of the solution (both for the constructive and the LNS) is very similar to the x variables of the MIP model. A binary matrix M with $\sum_{k \in K} C_k$ rows and $|T|$ columns, where rows $1...C_1$ represents vehicles of the first type of contract, $C_1 + 1...C_2$ the second, and so on. Their values represent the following:

$$M[i,j] = \begin{cases} 1 : \text{If the car of row } i \text{ is assigned to work on day } j \\ 0 : \text{otherwise} \end{cases}$$

An important auxiliary data structure is the v_ratio array,of length $|C|$. It holds the current ratio of vehicle types assigned on the solution. This array is updated every time a new vehicle is added or removed from the solution.

As state before, the FCAP has decisions on two dimensions: *i*-) the contract (vehicle type) that will be chosen and *ii*-) the days it will be assigned to work. Based on this we propose a greedy-randomized constructive heuristic, in order to find an initial feasible solution. It is *random* considering decision dimension *i*, and *greedy* considering *ii*. The procedure is described by Algorithm 3.

Algorithm 3: *Greedy-randomized constructive*

Data: D, C, U, P, F
Result: feasible solution M, cost Z

```
initialize_ds(M, D, U);
v_ratio ← update_ratio(M);
l_feasible ← check_ratio(v_ratio, P);
while ∑_{t∈T} D_t > 0 do
    v ← random_vehicle(l_feasible);
    sort(D);
    assign_vehicle(M, D, C, v);
    v_ratio ← update_ratio(M);
    l_feasible ← check_ratio(v_ratio, P);
    Z ← update_cost(Z, F, v);
```

Line 1 simply initialize all data-structures; note that, similar to the MIP model, we need to known upper-bounds on the number of vehicles (U) to determine the number of rows of M, so Algorithm 1 is also used here. Line 2 updates the array v_ratio with 0 (since there are no vehicles assigned, the ratio of each contract type is zero). With the current ratio a list of feasible types of contracts is updated (line 3). It checks which types of contracts (vehicles) did not surpassed their maximum allowed ratio (P).

With this set up, the vehicle assigning loop starts on line 4, and continues until there is no demand to be met on D. The first decision to be made is the type of vehicle to be assigned; recall that this is the random part of the algorithm, as shown in line 4. A vehicle type v is selected randomly according to the feasible list $l\_feasible$.

With the vehicle selected, the second decision is made: how to assign its working days. That is achieved in a greedy fashion, by first sorting the vector of demands D in a descending order (line 6), and starts by assigning days with the higher demand values (line 7), checking the coverage days (C) of that vehicle type. Every time a work day is assigned to the vehicle, the remaining demand is (D) is also updated.

After the assignment, the ratio array and the feasible list are updated (lines 9 and 10), and the solution cost increased (line 10).

Given the stochastic nature of the constructive algorithm, it is possible to run it several times and collect the best result. In the computational results we present this iterative version of the greedy constructive heuristic.

4.2 Destroy and repair operators

For the LNS we used destroy and repair operators that held similarities with the constructive procedure.

The destroy operator works as follows: at each iteration $\alpha\%$ of the assigned vehicles on the current solution are randomly removed (their work days are set to zero, and the remaining demand is increased again). After a number of non-improving iterations, the destroy factor α is increased, enlarging the size of the neighborhood.

A random-greedy repair operator is used, very similar to the constructive procedure: the destroyed solution is currently not feasible since there are unmet demand. Based on the $l\_feasible$ list, a random vehicle is selected and its work days assigned according to the sorted demand D. Although the repair operator acts in a similar manner as the constructive heuristic, it is shown in the results section (5) that is works very well for the FCAP.

5 Computational results

In this Section we present the computational results of the proposed algorithm, as well as the optimal values for the problem instances using the SCIP optimization solver.

Since the company currently performs the analysis using the software Microsoft Excel, and therefore all relevant data consists in spreadsheets, the authors have decided to implement the LNS heuristic in VBA (Visual Basic for Applications), for a seamlessly integration.

The LNS parameters used are showed in Table 1. We run the Greedy-randomized constructive **Greedy iterations** times before the LNS is started.

As previously mentioned, the SCIP solver was used to solve the mathematical programming model, with the default parameter settings.

Parameter	Value
Greedy iterations	50
LNS iterations	5000
Initial destroy factor	0.15
Destroy factor incremental step	0.05
Maximum destroy factor	0.40
Non-improving iterations until increase destroy factor	0.1 * (LNS iterations)

Table 1 LNS Parameters

Table 2 presents the results. All instances are from a set **T** of 31 days extracted from the company. Each instance has a **Mean demand**, which might indicate the complexity of the problem, as explained further in the text.

The GAP column is derived as:

$$\frac{|z(s) - z(s^*)|}{z(s^*)} \tag{9}$$

Where s is the LNS solution and s^* the optimal solution.

Mean demand	Instance	Z Optimal	Time SCIP	Z LNS	Time LNS	GAP
4.45	E10	10107	11	10192	2	0.84%
31.26	E11	52590	905	52592	48	0.00%
3.16	E12	4554	6	4963	1.5	8.98%
18.81	E14	19040	759	19063	10	0.12%
73.19	E15	116543	454	123543	67	6.01%
14.23	E16	23653	7	23936	3.5	1.20%
3.10	E17	6258	2	6294	1.57	0.58%
7.39	E18	7900	73	7900	2.8	**0.00%**
63.32	E19	92251	2369	92251	392	**0.00%**
26.45	E20	44124	591	45024	5	2.04%
9.84	E21	26155	66	26155	3	**0.00%**
6.00	E22	5676	47	5676	1.8	**0.00%**
60.68	E23	56133	47212	56392	355	0.46%
26.58	E24	33884	616	33922	34.3	0.11%
10.81	E25	37254	7	37261	5.8	0.02%

Table 2 Results (optimal x LNS)

We see from Table 2 that the LNS performs well on most instances, providing near optimal solutions (solutions with a GAP $< 1\%$) in 73.3% of all instances. Besides, the heuristic achieves those results much faster than the solver.

In order have a better sense of the time difference between the exact method and the heuristic, we plot the instances (ordered by increasing Mean demand values) and the time required for each method to solve it. This is the first plot in Figure 3.

We see from Table 2 that instance E23 have the longest computational time off all, and surpasses the second longest by far, making it act as an outlier when analyzing the plot. To

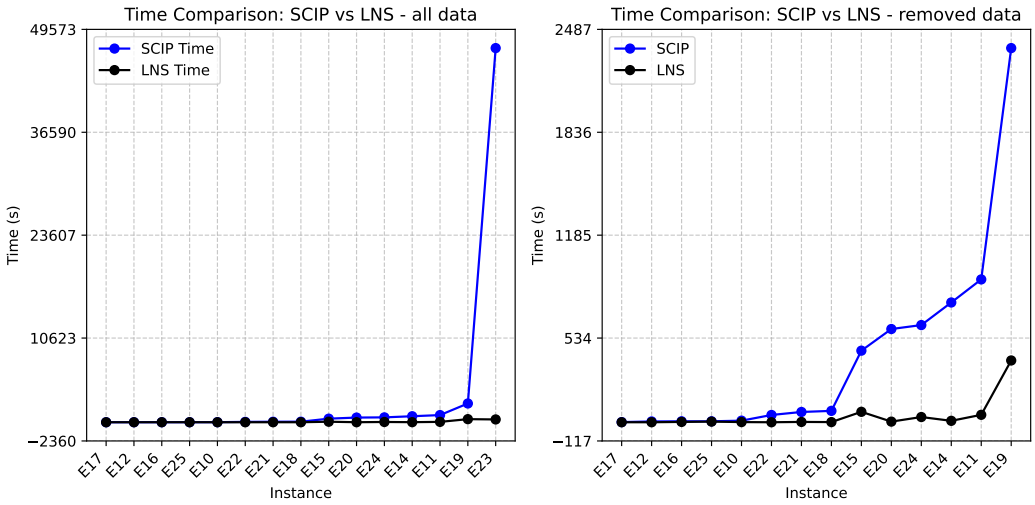

Figure 3 Time comparison plots: removed data and all data

better understand the time behavior, the second plot of Figure 3 have this instance removed. From the second plot it is apparent that starting on instance E15, the time required by the solver grows at a much greater scale than the heuristic procedure.

Also, since the problem instances were ordered by Mean demand, we see that this feature is the primary factor impacting the complexity of the algorithms to solve them. This behavior may suggest a somewhat hybrid general methodology on the solution of the problem, relying not solely on the heuristic, but rather using the solver on some complex cases: first classify the problem according to its Mean demand, and direct the solving method (SOLVER or heuristic) according to user parameters on time limit. In this way, the benefits of guaranteed optimal values are held on simpler instances, whereas the time saving provided by the LNS heuristic holds on complex ones.

6 Conclusions

In this paper we presented the Fleet Composition and Allocation Problem (FCAP), a logistic problem arising from a national-wide Brazilian company, regarding the lease decision of a fleet of vehicles. The lease decision is put forward trough contracts, which determine a set of days that a vehicle can be used. Different contracts entails different number of days to use the vehicle, and therefore different fixed costs. The allocation part of the problem rely on the decision of which days of the planning horizon the contracted vehicle will be used to perform deliveries.

An integer programming model is developed for the FCAP. In order to build the model there must exist an upper-bound on the number of vehicles available for each contract type. We developed an algorithm to extract that upper-bound from the instance parameters. As an alternative to the optimal solution provided by a solver (which may take more time than the company is willing to spent), we also implemented a LNS metaheuristic. The LNS is well fitted for problems were a neighborhood structure is not explicitly available, or a complete neighborhood exploration is too broad, which is the case of the FCAP.

Since the problem arose directly from the industry, in this paper we were interested in finding the best (if any) solution framework that could be seamlessly integrated onto the day-to-day activities of the personnel. That would, as a first attempt, exclude the use of commercial state-of-art solvers, like GUROBI (Gurobi Optimization, LLC (2024)) and CPLEX (Cplex (2009)), since the company do not already have any license for them.

We used the open-source SCIP solver (Bolusani et al. (2024)), which by the time of this writing is the best performing open-source solver available. As the programming language for the LNS implementation, the authors opted to maintain the some environment that the company currently uses to solve the problem - spreadsheets from Microsoft Excel. Therefore the metaheuristic was implemented in Visual Basic for Applications.

The computational results are promising for the company: the SCIP solver was able to find the optimal solution for all instances (the most time consuming took 13.11 hours). Even though this is not as nearly as fast as the company desires, it creates a baseline for future adoption of commercial solvers. Also, the results for the LNS metaheuristic were also satisfactory, providing near optimal solutions (solutions with a GAP $< 1\%$) in 73.3% of all instances demanding much less computational time. From the results we were also able to infer the main feature that brings complexity to an instance to be solved: the mean demand of vehicles - that is tied to the extraction of the upper-bound on the number of vehicles (both for the model and the data structures of the LSN).

Considering the computational results, a hybrid methodology can be adopted: when a new problem arrives, first classify it according to its mean demand value, and, depending on the result (and the user preference) a decision is made between the use of the solver or the LNS. This classification of the problem complexity could be made using a machine learning regression model, like regression trees. In this case, when the regression output is less than a user provided threshold, the solver is used. This idea was not implemented due to the lack of solved instances to train the model.

As a summary, this paper presents a new problem arising from a Brazilian industry, we proposed and tested solution methods that are on the reach of any logistic manager, since at they are based on open-source (SCIP) license, and a day-to-day software for most Windows users: Microsoft Excel. With the optimization method, managers would reduce the time required to decide on which contracts to lease, paving the way to more a more accurate and fast decision process.

References

Ahuja, R.K., Orlin, J.B., Sharma, D., 1998. New neighborhood search structures for the capacitated minimum spanning tree problem .

Ansaripoor, A.H., Oliveira, F.S., 2018. Flexible lease contracts in the fleet replacement problem with alternative fuel vehicles: A real-options approach. European Journal of Operational Research 266, 316–327.

Bolusani, S., Besançon, M., Bestuzheva, K., Chmiela, A., Dionísio, J., Donkiewicz, T., van Doornmalen, J., Eifler, L., Ghannam, M., Gleixner, A., Graczyk, C., Halbig, K., Hedtke, I., Hoen, A., Hojny, C., van der Hulst, R., Kamp, D., Koch, T., Kofler, K., Lentz, J., Manns, J., Mexi, G., Mühmer, E., Pfetsch, M.E., Schlösser, F., Serrano, F., Shinano, Y., Turner, M., Vigerske, S., Weninger, D., Xu, L., 2024.

The SCIP Optimization Suite 9.0. Technical Report. Optimization Online. URL: https://optimization-online.org/2024/02/the-scip-optimization-suite-9-0/.

Cplex, I.I., 2009. V12. 1: User's manual for cplex. International Business Machines Corporation 46, 157.

Dönmez, S., Koç, Ç., Altıparmak, F., 2022. The mixed fleet vehicle routing problem with partial recharging by multiple chargers: Mathematical model and adaptive large neighborhood search. Transportation Research Part E: Logistics and Transportation Review 167, 102917.

Efthymiadis, S., Liapis, N., Nenes, G., 2023. Solving a heterogeneous fleet multi-compartment vehicle routing problem: a case study. International Journal of Systems Science: Operations & Logistics 10, 2190474.

Golden, B., Assad, A., Levy, L., Gheysens, F., 1984. The fleet size and mix vehicle routing problem. Computers & Operations Research 11, 49–66.

Gurobi Optimization, LLC, 2024. Gurobi Optimizer Reference Manual. URL: https://www.gurobi.com.

Hoff, A., Andersson, H., Christiansen, M., Hasle, G., Løkketangen, A., 2010. Industrial aspects and literature survey: Fleet composition and routing. Computers & Operations Research 37, 2041–2061.

Li, W., Ou, J., 2024. Approximation algorithms for scheduling parallel machines with an energy constraint in green manufacturing. European Journal of Operational Research 314, 882–893.

Li, Y., Chu, F., Chu, C., Zhu, Z., 2019. An efficient three-level heuristic for the large-scaled multi-product production routing problem with outsourcing. European Journal of Operational Research 272, 914–927.

Martello, S., Toth, P., 1990. Knapsack problems: algorithms and computer implementations. John Wiley & Sons, Inc.

Pasha, U., Hoff, A., Hvattum, L.M., 2016. Simple heuristics for the multi-period fleet size and mix vehicle routing problem. INFOR: Information Systems and Operational Research 54, 97–120.

Shaw, P., 1998. Using constraint programming and local search methods to solve vehicle routing problems, in: International conference on principles and practice of constraint programming, Springer. pp. 417–431.

Utama, D.M., Widodo, D.S., Ibrahim, M.F., Dewi, S.K., 2020. A new hybrid butterfly optimization algorithm for green vehicle routing problem. Journal of Advanced Transportation 2020, 1–14.

Zäpfel, G., Bögl, M., 2012. Two heuristic solution concepts for the vehicle selection problem in line haul transports. European journal of operational research 217, 448–458.