A set-partitioning approach to the VeRoLog Solver Challenge 2019

Combinatorial Optimization

Group A1
Ania Serbina
Sammy Falvey
Jialing Huang
Leo Metasch
Ole Vriethoff

Supervisor: *Caroline Jagtenberg*

October 8, 2024

Abstract

In this paper, we report on our results to the problem posed by the VeRoLog Solver Challenge 2019. The problem consists of the delivery and subsequent installation of machines, two components that are strongly interlinked with each other. Furthermore, the problem contains a rich cost structure, technician skill sets and an implicit scheduling problem.

Our approach consists of decomposing the problem at hand into a machine delivery problem and an installation problem, which we both formulate as set-partitioning problems. We ensure interconnectedness of the two by adding formulations of idle days to the objective functions of both MIPs, after which we obtain solutions with Gurobi's solver. Besides laying the foundation of our approach, we report on its performance on several instances.

1 Introduction and literature review

The VeRoLog Solver Challenge 2019 was the 4th solver challenge organised by VeRoLog, which is an important working group within the Association of European Operational Research Societies. The challenge, which is organised every year, poses a complex vehicle routing problem to the participants and is aimed at promoting the development of efficient algorithms for real-world use cases. The 2019 edition concerned the distribution and subsequent installation of machines within a planning horizon of multiple days; for a comprehensive summary of the problem, we refer to the challenge's web page ("VeRoLog Solver Challenge 2019", 2019).

In this paper, we make an attempt at solving this problem. To this end, we first investigate prior academic work on solving similar problems, after which we will look into approaches to the 2019 Solver Challenge of several participants that performed well.

1.1 Vehicle Routing Problem

The Vehicle Routing Problem (VRP) is a well-known combinatorial optimization problem, with significant practical importance in areas such as logistics, transportation and delivery planning (Toth & Vigo, 2014). It was first introduced by Dantzig and Ramser (1959) as the Truck Dispatching Problem, which is a generalization of the Traveling Salesman Problem (TSP). The classical VRP consists of a class of problems in which a number of vehicles must serve customers with a certain demand. The vehicles start at the depot, serve the customers in their route and return to the depot after finishing their respective routes (Montoya-Torres et al., 2015).

The VRP problem has been extended in several ways, increasing its complexity in several aspects. As introduced by Dantzig and Ramser (1959), the basic capacitated VRP (CVRP) considers a homogenous fleet of vehicles with vehicle carrying capacity as a constraint. The VeRoLog Solver Challenge has several extensions to the CVRP. It considers several machine-requests that must be completed within certain time windows. The machines are delivered by an unlimited amount of homogeneous trucks, which are restricted by travel distance and carrying capacity. Each truck must start and end at the depot, where it is allowed to return to the depot during the day to pick up more machines. The truck problem is therefore a capacitated multi-trip multi-period VRP with a homogeneous fleet. At least one day after delivery, a machine is installed by a technician with a certain skill set, who starts and end at their home locations. Further restrictions are a maximum travel distance and a maximum number of installations per day. Because the skill sets and home locations vary per technician, this situation could be modeled as a capacitated multi-depot multi-period VRP with heterogeneous fleet.

1.2 Matheuristics

The term "matheuristic" was first coined in 2006 at the Workshop on Mathematical Contributions to Metaheuristics (Boschetti & Maniezzo, 2022). This conference was organised due to the observation that metaheuristics, though often creating good solutions to optimization problems, did not make use of the potential of mathematical programming approaches ("Matheuristics 2006: 1st Workshop on Mathematical Contributions to Metaheuristics", 2006). Matheuristics tend to combine the two; mathematical models serve as basis for developing heuristic algorithms. Next, we will discuss two interesting (categories of) matheuristics to the presented problem.

Formulating the problem as a set-partitioning problem is a common approach when solving (capacitated) VRPs. The set-partitioning problem approach to VRPs, which Doerner and Schmid (2010) describe as one of the major matheuristics, was first introduced by Balinski and Quandt (1964) on a problem that was slightly different from the standard VRP. Heuristic algorithms that took advantage of this formulation were subsequently deloped by Foster and Ryan (1976) and Cullen, Jarvis, and Ratliff (1981), after which Agarwal, Mathur, and Salkin (1989) proposed an exact algorithm to solving such problems. The latter described the set-partitioning formulation in the following simplified form:

$$\begin{aligned} \text{minimize} \quad & \sum_j c_j x_j \\ \text{subject to} \quad & \sum_j a_j x_j = e \\ & x_j \in \{0,1\},\ j = 1, \ldots, m \end{aligned}$$

In which c_j indicates the costs associated with route j and the elements $a_j i$ of vector a_j represent whether or not location i is visited by route j. By formulating the problem as above, it reduces to finding parts (tours) that together form the total set of points that have to be visited - this is here modelled by the all-ones vector e as right hand side for the first constraint.

Decomposition methods form another major matheuristic category in the context of VRPs according to Doerner and Schmid (2010) and Archetti and Speranza (2014). Decomposition entails breaking down a problem into subproblems, which can arise "naturally" from the independence of subproblems or can be done to approximate computationally hard problem by considering easier to solve subproblems (Jagtenberg et al., 2020).

1.3 Prior approaches

Limited research has been done on the exact routing problem posed by the VeRoLog Solver Challenge in 2019 before the challenge was organised (Gromicho, Van 't Hof, & Vigo, 2019). A close related but less complex version was formulated by Bae and Moon (2016), who also consider a delivery and subsequent installation problem, but this problem differs in terms of the planning horizon, cost structure and technician skill set.

When decomposing the problems into two subproblems, one of which considers the delivery of the machines by trucks and the other that takes account of the scheduling and routing of the technicians that install the machines, we arrive at two subproblems that closer resemble previously investigated problems. For example, the technician problem is similar to the service technician routing and scheduling problem (Graf, 2020). Additionally, according to Graf, the truck problem is comparable to the multi-period VRP with due dates (MVRPD), which has been considered by Francis, Smilowitz, and Tzur (2008) and Archetti, Jabali, and Speranza (2015).

After the challenge, several participants reported on the algorithms they used for their submissions. Jagtenberg et al. (2020) won the third price in the challenge by splitting the problem into a delivery problem and a technician installation problem, which they solved by developing

a new algorithm. This algorithm, which is called "columnwise neighborhood search", is a column generation procedure for problems that are formulated as set-partitioning problems. By using this procedure, the size of a problem can be expanded gradually by looking into promising neighboring parts.

Another well-performing approach was conducted by Kastrati, Ahmeti, and Musliu (2020), who used an iterative local search combined with destroy and repair heuristics. The first prize was won by Graf (2020), who also decomposed the problem into a truck delivery problem and a technician routing/scheduling problem, which he subsequently solved using an approach consisting of a combination of Large Neighborhood Search (LNS) and local search heuristics.

2 NP-Completeness

The presented problem is a complex combination of two vehicle routing problems, which must be solved in combination to obtain a feasible solution. This problem is an NP-complete problem. Two steps are required to prove that this problem is indeed an NP-complete problem. The first is to prove that a solution that was found for this problem can be verified to be feasible in polynomial time. The second step is to prove that any problem in NP can be reduced to this problem in polynomial time.

To check if solutions can be verified in polynomial time, it is necessary to consider what a verification algorithm would need to check. In this case, the algorithm has to check that none of the imposed constraints are violated. This can be done in two steps. First, the delivery schedule needs to be verified in the following steps: All requests need to be delivered within their specified time window. All truck routes start and end at the depot, and all truck routes do not exceed the truck capacity or distance limit.

Second, verify the installation schedule, that all requests are installed, that all installations occur at least one day after the delivery of a request, and that all requests are installed by a technician with an appropriate skill set. Next, check that all technicians comply with their daily distance and installation limits and they all adhere to their mandatory work schedule.

Each of these checks can be completed in polynomial time with respect to the input size, which in this case is the number of requests, technicians, and trucks. Therefore, a solution can be verified in polynomial time, and the problem is in NP.

The simplest way to prove this problem to be NP-Hard is to find a problem that is NP-Hard and reduces to this problem. In this case, the Travelling Salesman Problem (TSP) is chosen as it is most closely related to the problem at hand. To reduce the problem, any instance of the TSP must be transformed into an instance of the VSC. First, every city in the TSP becomes a customer location, with the exception of the starting city, which becomes the depot. Next, one request with one machine of a single type is created for each customer location. The delivery time window for each request is set as the entire planning horizon. The truck capacity and distance limitations are set high enough to visit every request and return to the depot. Initialize technicians with distance and capacity constraints that allow them to serve all requests.

Next, set the costs for technicians, technician days, technician distance, trucks, truck days, and idle days equal to zero, and set the truck distance cost equal to the distance cost between cities in the TSP instance.

The only cost in this version of the VSC is the truck distance cost. In this case, the overall travel distance of the trucks will be minimized, resulting in the optimal solution for this instance and also the optimal solution in the TSP instance. As solving the VSC also solves TSP, TSP reduces to the VSC, and the VSC is NP-Hard.

From the two-part proof, it follows that the VSC is in NP and is NP-hard. Therefore, it is NP-complete.

3 Methodology

In this section, we explain how we matheuristically modeled this problem in Python. The required Python packages to run our solution are gurobipy from Gurobi 11.0.1 and numpy.

3.1 Technician Problem

To make the problem computationally less demanding, we decide to decompose it into two subproblems: a truck problem, which handles the delivery of the machines and a technician problem, which handles the scheduling of technicians and their routing for the subsequent installations. We first look to a solution for the technician subproblem, because it is the most complex. After this, the obtained technician schedule is used as an input for solving the truck problem.

We decide to make use of the set-partitioning approach of Jagtenberg et al. (2020). To this end, we need to define both of our subproblems as set-partitioning MIPs, in which the parts consist of the routes for either the technicians or trucks. First, these routes need to be created.

For the technician problem, we decide to create all possible tours with maximum length 5. This number is chosen to keep the computation time limited. Furthermore, we noticed that the instances rarely allow technicians to serve more than 5 requests. As the solver will assign the possible tours to technicians, we choose to generate general tours irrespective of the technician. As such, only the distances between the requests are considered, and we only keep the routes that form a shortest round of the visited requests; not feeding the other possible orders of requests to the solver yields a significant reduction in the solution space. Furthermore, the routes with length larger than the longest distance restriction of the available technicians are deleted from the candidate routes.

The technician tours that remain form the candidate parts for the technician set-partitioning problem, which is solved using Gurobi's solver. The MIP of the subproblem is displayed below:

Constants and Decision Variables:

- b_{tm}: 1 if tour t includes machine m, 0 otherwise
- h_{pt}: cost of person p completing tour t (includes TECHNICIAN_DAY_COST and TECHNICIAN_DISTANCE_COST)
- e_{sd}: 1 if schedule s contains day d, 0 otherwise
- l_{md}: 1 if machine request m can be installed on day d, 0 otherwise
- n: number of unique technicians hired
- y_{ptd}: 1 if person p performs tour t on day d, 0 otherwise
- z_{ps}: 1 if person p has schedule s, 0 otherwise
- w_m: number of certain idle days for machine m

Minimize:

$$\sum_{p \in P} \sum_{t \in T} \sum_{d \in D} h_{pt} \cdot y_{ptd} + \sum_{m \in M} \big(w_m \cdot \text{IDLE\_DAY\_COST[m]}\big) + n \cdot \text{TECHNICIAN\_COST}$$

Subject to:

$$\sum_{s \in S} z_{ps} = 1 \quad \forall p \in P \tag{1}$$

$$\sum_{t \in T} y_{ptd} \leq \sum_{s \in S} e_{sd} \cdot z_{ps} \quad \forall p \in P, \forall d \in D \tag{2}$$

$$\sum_{p \in P} \sum_{t \in T} \sum_{d \in D} y_{ptd} \cdot b_{tm} = 1 \quad \forall m \in M \tag{3}$$

$$\sum_{p \in P} \sum_{t \in T} b_{tm} \cdot y_{ptd} \leq l_{md} \quad \forall m \in M, \forall d \in D \tag{4}$$

$$\sum_{m \in M} b_{tm} \cdot y_{ptd} \leq \text{MAX\_REQUESTS[p]} \quad \forall p \in P, \forall t \in T, \forall d \in D \tag{5}$$

$$\sum_{p \in P} \sum_{t \in T} \sum_{d \in D} (d \cdot b_{tm} \cdot y_{ptd}) - \text{LAST\_DELIVERY\_DAY[m]} - 1 \leq w_m \quad \forall m \in M \tag{6}$$

(1)

In this MIP, S is the set of feasible schedules. Next to the distance, capacity, and schedule constraints given by the challenge, we implemented a restriction on the decision variable l_{md} in constraint (4), which enforces that installations may not be done before a certain date. It is evident that the installation can not start before the day after the first request day; it is, however, also possible to give the technician problem less room and use the earliest starting date for technicians as a parameter. We decide to run every instance twice with the restriction to install only the day after the delivery window opens and the day it closes, as these two options are the only ones to produce optimal results for the given instances. Another addition is made to the objective function: the idle costs that are always incurred if the installation happens after the delivery window closes are included in the objective, which leads to the model preferring delivery dates without guaranteed idle costs. Adding this constraint strengthens the connection between both subproblems.

3.2 Truck Problem

In the second phase of the algorithm, after obtaining the optimized schedule for the installations of every request, the delivery schedule for the trucks is optimized. The truck MIP works similarly to the technician MIP, and again feasible routes with a maximum length of 5 locations are created. This time, however, the tours are allowed to visit the depot in between two delivery requests. The truck MIP is implemented in Gurobi as illustrated below; the MIP enforces all capacity and distance restrictions and adds the restriction limiting the delivery days based on the technician problem solution.

- x_{rd}: 1 if tour r performed on day d, 0 otherwise
- f_d: number of trucks used on day d
- a_{rm}: 1 if route r contains request m, 0 otherwise
- w_m: number of idle days for request m
- n: number of trucks used

Minimize:

$$\sum_{r \in R} \sum_{d \in D} \Big(x_{rd} \cdot l_r \cdot \text{TRUCK\_DISTANCE\_COST} \Big) + \sum_{d \in D} \Big(f_d \cdot \text{TRUCK\_DAY\_COST} \Big) + \sum_{m \in M} \Big(w_m \cdot \text{IDLE\_DAY\_COST[m]} \Big) + n \cdot \text{TRUCK\_COST}$$

Subject to:

$$\sum_{r \in R} \sum_{d \in D} \left(a_{rm} \cdot x_{rd}\right) = 1 \quad \forall m \in M \tag{1}$$

$$\sum_{r \in R} x_{rd} \leq n \quad \forall d \in D \tag{2}$$

$$\sum_{r \in R} x_{rd} = f_d \quad \forall d \in D \tag{3}$$

$$\text{INSTALL\_DAY[m]} - \sum_{r \in R} \sum_{d \in D} \left(d \cdot a_{rm} \cdot x_{rd}\right) - 1 = w_m \quad \forall m \in M \tag{4}$$

$$\sum_{d \in \text{DELIV\_WIND[m]}} \sum_{r \in R} \left(a_{rm} \cdot x_{rd}\right) = 1 \quad \forall m \in M \tag{5}$$

$$\sum_{d=1}^{\text{INSTALL\_DAY[m]}} \sum_{r \in R} \left(a_{rm} \cdot x_{rd}\right) = 1 \quad \forall m \in M \tag{6}$$

4 Results

4.1 Objective values

Instance	Objective value	% above best-in-class	Total run time	Technician % run
1	255,690	0.25%	20.09	95.0%
2	277,955	2.52%	4.22	78.8%
3	438,388	0.33%	5.61	69.6%
4	28,825	0.37%	6.71	87.5%
5	12,975	3.18%	14.11	85.4%
6	320,538	2.33%	531.03	86.0%
7	9,263,455	6.02%	2,382.38	97.3%
8	8,364,170	0.02%	491.67	90.4%
9	158,725	7.06%	604.02	92.2%
10	802,140	28.57%	1,972.87	96.3%
11	675,390,880	0.02%	1,550.10	93.2%
12	95,289,614	0.12%	335.93	71.4%
13	310,165	1.82%	236.95	96.3%
14	1,304,427	3.07%	130.83	95.3%
15	98,684,821	9.21%	329.00	92.3%
16	1,215,710	10.55%	2,653.53	97.4%
17	164,497,650	36.97%	2,490.63	96.0%
18	110,648,625	4.92%	781.44	88.6%
19	507,369	1.75%	1,282.11	95.9%
20	371,230	1.46%	2,709.80	90.6%

Table 1: Objective values and run time (seconds) per instance

4.2 Analysis of Results

As observed in Table 1, the algorithm performs relatively well, as the objective value of 14 instances is less than 5% above that of the best-in-class solution (as of 20 May 2024 10.00 AM). This is in general true for the smaller as well as the larger instances.

Our worst-performing solutions are that of instance 17 (36.97% higher), 10 (28.57% higher) and 15 (14.84% higher). For instance 17, it is likely that our maximum tour length (the maximum number of stops in a tour) of 5 is too short, because the maximum number of installations of the utilised technicians are 5, 8 and 10. An increase in tour length could improve our solution significantly, because of the high relative technician distance costs of this instance. A similar extension would likely improve the solution of instance 15, which currently has three technicians with a maximum of 5, 6 and 7 installations and three with a maximum of 8 scheduled. For instance 10, we use one technician which can perform all type of jobs and has a maximum of 5 installations per day. 3 other technicians have a similar skill set but higher maximum of installations. Since the technician travel costs are relatively low compared to the other technician costs, longer routes are a relative cheap solution to improve our results.

Our best-performing solutions are that of instance 8, 11 and 12. Instance 11 has relative high technician distance costs, which makes shorter routes more desirable. Our algorithm performs in particular well in cases where TRUCK_DISTANCE_COST is considerably higher than the TECHNICIAN_DISTANCE_COST. For instance 8 and 12, $\frac{TRUCK\_DISTANCE\_COST}{TECHNICIAN\_DISTANCE\_COST} = 100$, and for instance 11 it is 10, much higher than that of the other instances (for which this ratio equals 2 at most). We also observe that instance 11 and 12 are among the larger instances. This is also observed in the largest instance (20), which performs particularly well in the total ranking.

4.3 Randomness and run time

We observed that running the algorithm multiple times (at least twice per instance) leads to the exact same solution. Therefore, it seems like our algorithm does not involve randomness in terms of obtaining the optimal solution. A curiosity we have noticed is that even though the results are the same, they can differ when run on different computers, an issue that we could not attribute since it is not different for all computers. Our current algorithm would not achieve better results when running it longer, but some improvement heuristics, which would increase our run time, could.

For the run time, we did observe variations when running each instance multiple times. However, the ratio of the running time is always dominated by the technician problem as it considers all possible routes instead of a subset.

4.4 Improvement heuristics

The main improvement heuristic of our algorithm seems to be an increase of the maximum tour length. This measure will increase the solution space, but also increases the run time in the same way. There is a trade-off between the computation time and cost savings for the tour length. In particular when technicians have high travel costs and maximum of installations, the maximum tour length should be increased.

5 Conclusion & Discussion

In this paper, we presented our algorithm that we used to solve the problem posed by the 2019 VeRoLog Solver Challenge. Our approach of decomposing the problem into two subproblems, formulating them as ILPs and then solving them using Gurobi yields promising results for the given instances and performs well compared to other solutions in the leaderboard. However, when analyzing the algorithm laid out in the methodology, several directions for improving our solutions can be identified.

For example, as the current algorithm splits the problem into two components - the technician and the truck problem - the instance can never be optimized in its entirety. Optimizing

the technician problem first and imposing this solution on the Truck problem leads to limited optimization possibilities in the combination of the problems. This approach is advantageous for larger instances as the time required to optimize is shorter, but capturing both problems into one MIP will almost certainly result in better solutions for the given instances.

The other possible directions of improvement have been identified in the tour generation for the technician problem. The first shortcoming of the current tour generation method is the assumption that the same order of serving a combination of requests is the same for every technician. This can lead to combinations that would usually be assigned to a technician, not being feasible or at least not optimal with the current order that is set when feeding the generated routes into the MIP. An approach for dealing with this shortcoming would be to make the route generation more based on the individual technicians and consider a different ordering of requests within a tour for each technician. This, however, is very difficult given the current configuration of the algorithm.

This leads to the second direction of improvement regarding the tour creation for technicians: the current algorithm works with the restriction that a technician tour can only have a maximum of five stops per tour, as the creation of longer tours and the following optimization with the much higher number of tours does not conclude in a reasonable time frame. A possible improvement could be achieved by making use of construction heuristics, e.g. Sweep, to create longer tours and give technicians more options while keeping the running time reasonable.

These three directions are the major shortcomings of the applied algorithm. Other improvements might be achieved by varying the solution found using the algorithm and running a local search algorithm; the improvement effect of this, however, is expected to be minor due to the large size of the solution space.

To conclude, the algorithm in this report is well suited to optimize the Verolog Solver Challenge 2019. It has a reasonable runtime, and while it has some shortcomings in some instances, the overall performance is promising, especially with larger instances. It, therefore, is an overall good model for this problem.

6 Roles and Responsibilities

All should receive the same grade.

References

Agarwal, Y., Mathur, K., & Salkin, M. (1989). A set-partitioning-based exact algorithm for the vehicle routing problem. *Networks*, *19*, 731–749. https://doi.org/10.1002/net.3230190702

Archetti, C., Jabali, O., & Speranza, M. G. (2015). Multi-period vehicle routing problem with due dates. *Computers Operations Research*, *61*, 122–134.

Archetti, C., & Speranza, M. (2014). A survey on matheuristics for routing problems. *EURO Journal on Computational Optimization*, *2*(4), 223–246. https://doi.org/https://doi.org/10.1007/s13675-014-0030-7

Bae, H., & Moon, I. (2016). Multi-depot vehicle routing problem with time windows considering delivery and installation vehicles. *Applied Mathematics and Computation*, *40*(13), 6536–6549.

Balinski, M. L., & Quandt, R. E. (1964). On an integer program for a delivery problem. *Operations Research*, *12*(2), 300–304. Retrieved May 20, 2024, from http://www.jstor.org/stable/167930

Boschetti, M. A., & Maniezzo, V. (2022). Matheuristics: Using mathematics for heuristic design. *4OR*, *20*(2), 173–208. https://doi.org/10.1007/s10288-022-00510-8

Cullen, F. H., Jarvis, J. J., & Ratliff, H. D. (1981). Set partitioning based heuristics for interactive routing [125]. *Networks*, *11*(2), 125–143. https://doi.org/10.1002/net.3230110206

Dantzig, G. B., & Ramser, J. (1959). The truck dispatching problem. *Management Science*, *6*, 80–91.

Doerner, K., & Schmid, V. (2010). Survey: Matheuristics for rich vehicle routing problems [206]. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *6373 LNCS*, 206–221. https://doi.org/10.1007/978-3-642-16054-7_15

Foster, B. A., & Ryan, D. M. (1976). An integer programming approach to the vehicle scheduling problem. *Operational Research Quarterly (1970-1977)*, *27*(2), 367–384. Retrieved May 20, 2024, from http://www.jstor.org/stable/3009018

Francis, P. M., Smilowitz, K. R., & Tzur, M. (2008). The period vehicle routing problem and its extensions. https://doi.org/10.1007/978-0-387-77778-8_4

Graf, B. (2020). Adaptive large variable neighborhood search for a multiperiod vehicle and technician routing problem. *Networks*, *76*(2), 256–272. https://doi.org/https://doi-org.vu-nl.idm.oclc.org/10.1002/net.21959

Gromicho, J., Van 't Hof, P., & Vigo, D. (2019). The verolog solver challenge 2019. *Journal On Vehicle Routing Algorithms*, *2*(1–4), 109–111. https://doi.org/10.1007/s41604-019-00011-8

Jagtenberg, C. J., Maclaren, O. J., Mason, A. J., Raith, A., Shen, K., & Sundvick, M. (2020). Columnwise neighborhood search: A novel set partitioning matheuristic and its application to the verolog solver challenge 2019. *Networks*, *76*(2), 273–293. https://doi.org/https://doi-org.vu-nl.idm.oclc.org/10.1002/net.21961

Kastrati, V., Ahmeti, A., & Musliu, N. (2020). Solving vehicle routing and scheduling with delivery and installation of machines using ils. https://api.semanticscholar.org/CorpusID:236908284

Matheuristics 2006: 1st workshop on mathematical contributions to metaheuristics. In: 2006. http://astarte.csr.unibo.it/Matheuristics2006/

Montoya-Torres, J. R., López Franco, J., Nieto Isaza, S., Felizzola Jiménez, H., & Herazo-Padilla, N. (2015). A literature review on the vehicle routing problem with multiple depots. *Computers & Industrial Engineering*, *79*, 115–129.

Toth, P., & Vigo, D. (2014). Vehicle routing: Problems, methods, and applications.

Verolog solver challenge 2019. (2019). https://verolog2019.ortec.com