COMPARISON OF CAPABILITIES OF RECENT OPEN-SOURCE TOOLS FOR SOLVING CAPACITATED VEHICLE ROUTING PROBLEMS WITH TIME WINDOWS

Marek KARKULA, Jerzy DUDA, Iwona SKALNA

AGH University of Science and Technology, Cracow, Poland, EU,
mkarkula@zarz.agh.edu.pl, jduda@zarz.agh.edu.pl, iskalna@zarz.agh.edu.pl

Abstract

Strong competition in the logistics industry requires increasingly better vehicle route planning (VRP). Not surprisingly, for several years we can observe increasing interest in developing optimization tools for solving real logistics problems. This is supported by the increase in computing power of personal computers and the fact that advanced optimization solvers are often developed as free or open source software. The question arises about the quality of solutions produced by such solvers and whether they can be useful for solving real life problems. In this paper, we compare the quality of solutions and the computational cost of the most popular libraries (OR-Tools, VROOM and jsprit) for solving the Capacitated Vehicle Route Planning with Time Windows (CVRPTW) problem, which is one of the most common problems in logistics environment. The comparison is performed on 56 benchmark problems described in the literature with known optimal solutions.

Keywords: Capacitated Vehicle Routing Problem with Time Windows, OR-Tools, VROOM, jsprit

1. INTRODUCTION

Transport process and delivery planning are among the most important tasks of managers in distribution, trade and production companies. The problem of route planning concerns the rationalization of distribution processes of products offered by the company's network of customers. Even though it has been studied for years, the research around it is still very active.

In operational research, route planning problem is included in the class of vehicle routing problems (VRPs). The vehicle routing problem is a combinatorial optimization and integer programming problem. It generally asks for a set of routes (see **Figure 1**) that satisfy constraints and minimize global transportation cost (could be monetary or distance or any other).

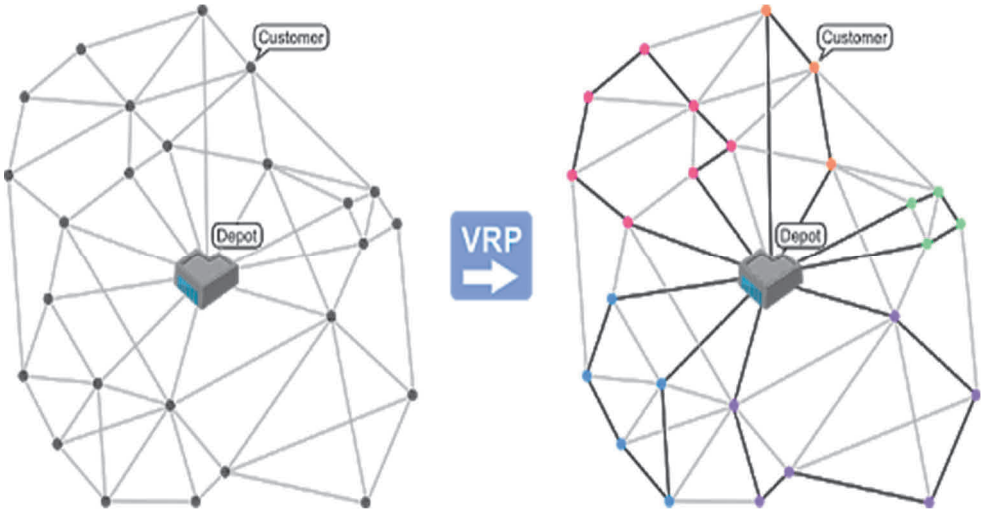

Figure 1 Vehicle Route Planning problem with single depot

VRP can be affected by various factors, just to mention vehicle capacity, number of depots, traffic conditions or time restrictions (e.g., imposed time windows). This makes routing and scheduling optimization very difficult. In fact, VRP is an NP-hard (non-deterministic polynomial-time hard) problem [1], which means that, in practice, it is not possible to build an algorithm that will always provide the optimal solution in an acceptable amount of time.

However, recent advances on Information and Communications Technologies (ICT) - growing use of GPS, distributed systems, new computing technologies - open new possibilities for optimizing the planning process of road transportation [2]. New technologies combined with advanced simulation and optimization techniques allow the practical development and implementation of new ICT-based solutions to support decision-making in the transportation and logistics arena. The increase in computing power and the development of new heuristic algorithms have enabled the development of efficient tools capable of solving many transport issues, primarily those that can be represented in the form of VRP.

In this paper we focus on open-source software for solving VRPs. We compare three popular toolkits that can handle additional constraints such as heterogeneous fleet, capacity limits or time windows. The first tool we consider (Section 3.1) is VROOM - a relatively new tool, written in C++ and presented in 2018, primarily developed for solving logistic problems that can be presented using OpenStreetMap. Next tool, discussed in Section 3.2, is jsprint - a popular package developed in Java environment since 2016 and distributed on LGPL license. Finally, in Section 3.3, the OR-Tools suite is reviewed, which is currently being developed by Google and is gaining great popularity as an efficient, universal optimization package for various combinatorial problems. All presented tools are compared in Section 4 by using 56 Solomon benchmark problems. The paper ends with some concluding remarks.

2. CAPACITATED VEHICLE ROUTE PLANNING WITH TIME WINDOWS

Depending on which factors are considered, one can distinguish Capacitated VRP (CVRP) with limits on the capacity of the vehicles in a fleet, Multi Depot VRP (MDVRP) with multiple depots (there is no need to return to the starting depot), Dynamic VRP (DVRP) where routes can change dynamically, Time Window VRP (TWVRP) with limits on the delivery time, Heterogenous VRP (HVRP) which allows different types of vehicles in the fleet, Time Dependent VRP (TDVRP) with time dependent travel times (time employed to traverse a route depends on the time of the day the travel starts from its originating node) or Green VRP (GVRP) where the transportation fleet is composed of electric vehicles with limited autonomy in need of recharge during their duties. Each of these problems is of great importance for the development of the industry, so there is a need for software that can combine majority or even all the above VRP variants and provide an optimal solution. In further considerations, however, we will focus on Capacitated VRP with time windows (CVRPTW), which is usually used as a core in the so-called rich VRP that usually include a heterogeneous fleet and other constraints, e.g., drivers' working time [3].

The capacitated vehicle routing problem (CVRP) was introduced by Dantzig and Ramser [4]. The basic concept of CVRP is to find a feasible set of vehicle routes that minimizes the total traveling distance and the total number of vehicles used. In each route, the vehicle departs from a given depot and returns to the same depot after completing service. Classical CVRP involves a single depot, homogeneous fleet of vehicles with limited capacity, and a set of customers who require delivery of goods from the depot. CVRP with time windows (CVRPTW) impose additional constraint that the goods must be delivered within specified time windows. Let us notice that finding even a feasible solution to CVRPTW with a homogenous fleet is a NP-complete problem [5]. CVRPTW is one of the most practical VRP problems, therefore several benchmarks sets have been defined for this problem in order to assess the quality of dedicated solution algorithms. The most frequently used benchmarks for CVRPTW are the ones generated by Solomon [6] and later by Gehring and Homberger [7]. In our research we focus on the first benchmark set as it is the most popular.

3. SELECTED OPEN-SOURCE SOLVERS

There are a number of barriers related to the development of your own algorithms to solve practical VRP class problems, whereas the commercial software has its source code closed, therefore, further development of the systems based on them requires close cooperation with the software supplier. As it was mentioned in the introduction, a number of open source tools and libraries are now available to resolve various types of VRP. The most popular examples of currently and actively developed open-source projects on VRP solving tools are, among others, OR-Tools, VROOM, jsprit, Open-VRP, OptaPlanner, SYMPHONY, and VRP Spreadsheet Solver. However, choosing the most suitable library to solve a particular practical vehicle routing problem is not easy since those libraries differ in terms of speed, programming environment, flexibility, functionality and ease of use. Several criteria may be used to assess their applicability in real world scenarios:

- generality - the possibility of solving various VRP types,
- flexibility in choosing and/or adding own construction algorithm (in order to generate an initial solution) and own repair algorithm (heuristic and/or meta-heuristic),
- flexibility in defining objective function - rigid and flexible (defined by a user),
- flexibility in defining constraints - the possibility of taking into account both “rigid” and “soft” constraints,
- execution speed, especially for large problem instances,
- quality of produced solutions,
- integration abilities with external systems (TMS, ERP, WMS, and others) as well as GIS and visualization systems,
- testing abilities - the possibility of using benchmark sets and different datasets of various formats (csv, xls/xlsx, json, xml, sql and others),
- availability of API documentation and its quality.

In what follows, the OR-Tools, VROOM and jsprit toolkits are discussed in details. The reason for choosing these three toolkits is that in our opinion they seem to be the most promising, among the available software, for solving various VRP problems. They are relatively young and are still intensively developed.

3.1. VROOM

Vehicle Routing Open-source Optimization Machine (VROOM) is an open-source software written in C++ to solve vehicle routing problems (VRP) arising in logistics and more widely in any context with geographically distributed tasks (source is available from site https://github.com/VROOM-Project/vroom).

VROOM uses several heuristics to find an initial solution depending on the problem: an adjusted version of the Christofides heuristic for TSP, clustering heuristics using spanning trees (for CVRP), adjusted versions of the Solomon insertion heuristics (for VRPTW and CVRP). Then a local search procedure is used to check for valid neighboring solutions and improve the current solution. The local search procedure uses 14 different operators that include, e.g., TwoOpt, various types of exchange and reallocate operators. The authors describe them in [8].

3.2. JSPRIT

Jsprit is a java based, open source (released under Apache License v2, available from https://github.com/graphhopper/jsprit) toolkit for solving rich traveling salesman and vehicle routing problems. It is lightweight, flexible and easy-to-use. jsprit can solve problems with pickups and deliveries, back hauls, heterogeneous fleets, finite and infinite fleets, multiple depots, time windows, open routes, different start and end locations, multiple capacity dimensions, initial loads, skills, etc.

The heuristic behind jsprit is based on ruin and recreate approach used in the large neighborhood search (LNS) proposed originally by Shaw [9] and adopted by Shimp et al. [10] to VRP. Shimp et al. combined the idea of simulated annealing with some threshold-accepting algorithms. In the ruin step an initial solution is disintegrated into two sets: one containing points that are no longer visited by a given vehicle and the second containing the remaining points that constitutes a partial solution. In the recreate phase points not assigned to any vehicle are reintegrated into the partial solution creating a new solution. The ruin and recreate steps are repeated until a stopping criterion is met (usually number of iterations).

3.3. OR-TOOLS

OR-Tools is Google's open-source, fast and portable software suite for solving combinatorial optimization problems (also available from github repository: https://github.com/google/or-tools). It supports various programming languages, including C++, C#, Java and Python. OR-Tools can solve many types of VRPs, including problems with pickups and deliveries, heterogeneous fleets, multiple depots, time windows, different start and end locations, multiple capacity dimensions, initial loads, skills, etc.

Obviously, there are some limitations on solving VRPs as they are inherently intractable for larger instances. Therefore, OR-Tools sometimes returns solutions that are good, but not optimal. Better solutions can sometimes be found by changing the search options for the solver. The first solution strategy is used to find an initial solution (one can also specify a set of already found initial routes). There are 13 various first solution strategies including AUTOMATIC and PARALLEL_CHEAPEST_INSERTION. The AUTOMATIC strategy lets the solver detect the strategy to be used according to the model being solved, whereas the PARALLEL_CHEAPEST_INSERTION strategy iteratively build a solution by inserting the cheapest node at its cheapest position; the cost of insertion is based on the arc cost function. Each of the 13 strategies can be incorporated into 6 different heuristics, such as greedy descent, simulated annealing, and tabu search.

OR-Tools documentation is available with plenty of examples. However, when referring to the API docs (which are presented in C++) there appears to be certain things that are either not represented in, e.g., the Python API, or not well documented to understand how it should be used.

4. NUMERICAL EXPERIMENTS

In this section we compare the described above optimization tools using six sets of CVRPTW benchmark problems designed by Solomon [6]. These problem instances were generated so that to highlight various factors that affect the behavior of routing and scheduling algorithms for CVRPTW. The geographical data are random in problem sets R1 and R2, clustered in problem sets C1 and C2, and a mixture of random and clustered structures in problem sets by RC1 and RC2. Problem sets R1, C1 and RC1 have a short scheduling horizon and allow only a few (approximately 5 to 10) customers per route, whereas the sets R2, C2 and RC2 have a long scheduling horizon permitting many customers (more than 30) to be serviced by the same vehicle. The customer coordinates are identical for all problems within one type (R, C and RC), whereas the time windows vary from very tight ones with a width of 10 time units to the ones that are hardly constraining with a width of 90 time units. For all the problems it is assumed that there is only one depot and the fleet is homogenous. The larger problems contain 100 customers to be served. Travel times of vehicles are equal to the corresponding distances, which are calculated based on the Euclidean distance between points. We have performed the calculations using each of the compared methods for all instances, i.e., 168 computational experiments were carried out in total. The average distance, average utilization of the vehicles and average running time for the obtained results in each of the Solomon's set are presented in **Table 1**.

Based on the obtained results, it can be concluded that on average jsprit achieves the shortest total distances travelled by vehicles. However, it requires much more computational time when compared to the other two solvers, albeit still acceptable in business applications (ca. 2 minutes). If the waiting time for a solution is

compared, the fastest is OR-Tools with a search heuristic set to automatic. Unfortunately, this is done at the expense of much worse results compared to other algorithms. Taking into account both the execution time and the quality of the results it seems that the most promising among the considered solvers is VROOM. The distances produced by VROOM are better than distances produced by OR-Tools and only slightly worse than distances produced by jsprit, whereas the computational times for VROOM are comparable with OR-Tools times and much they are better than jsprit times.

Table 1 Average results (distance, vehicles used and CPU times) for 56 Solomon benchmark instances with 100 customers; best results are marked with bold type

Case	VROOM			jsprit			OR-Tools		
	distance	vehicles	CPU[s]	distance	vehicles	CPU[s]	distance	vehicles	CPU[s]
C1	829.69	10	0.47	**828.38**	10	109.11	868.82	10	**0.38**
C2	592.08	3	0.98	**589.86**	3	85.36	621.95	4	**0.51**
R1	1241.39	14	0.82	**1204.56**	14	133.28	1283.56	15	**0.57**
R2	935.14	5	2.17	**890.42**	5	114.00	980.65	7	**0.65**
RC1	1395.60	14	0.77	**1348.51**	13	124.24	1459.49	15	**0.62**
RC2	1065.93	5	2.17	**1014.14**	6	115.23	1087.47	8	**0.77**

Table 2 shows the instances with the largest and the smallest differences between the results produced by the considered optimization tools. Additionally, best known (BK) results are compared with jsprit results.

Table 2 Instances with the largest and the smallest differences between the results of optimization tools

Instance	VROOM	ORtools	jspirit	BK	VR-js (%)	OR-Js (%)	js-BK (%)
C101	828.94	828.94	828.94	827.30	0.0	0.0	0.2
C104	832.78	1071.35	824.77	822.90	1.0	29.9	0.2
C201	591.56	737.04	591.56	589.10	0.0	24.6	0.4
C202	591.56	593.00	591.56	589.10	0.0	0.2	0.4
R101	1666.20	1689.90	1646.21	1637.70	1.2	2.7	0.5
R109	1203.63	1340.87	1152.38	1146.90	4.4	16.4	0.5
R203	901.63	945.64	876.25	905.72	2.9	7.9	-3.3
R211	846.69	902.26	766.63	761.10	10.4	17.7	0.7
RC102	1511.65	1574.45	1480.55	1466.84	2.1	6.3	0.9
RC103	1341.96	1458.28	1274.74	1261.67	5.3	14.4	1.0
RC204	819.03	823.08	789.07	798.46	3.8	4.3	-1.2
RC208	834.98	976.50	796.35	828.14	4.9	22.6	-3.8

As we can see from **Table 2**, for C1 and C2 benchmark sets the differences between VROOM and jsprit results are negligible. For some instances from those sets, OR-Tools produced very similar results, however for the other instances the OR-Tools results are quite distant from the remaining ones, and the difference reaches more than 20 %. For R1 and R2 sets the difference between VROOM and jsprit is more than 1 %, up to 10 %, while for OR-Tools the difference is at best 2.7 % and can reach up to 17.7 %. For some instances jsprit was able to generate the solution better than described in literature (for metaheuristics) in terms of total distance,

but for a cost of more vehicle used. The differences for the last group of instances RC1 and RC2 are even higher for OR-Tools and can reach more than 22 %, while for VROOM the difference with jsprit does not exceed 5.3 % (although starts from 2.1 %).

5. CONCLUSION

We have investigated the usefulness of some popular open-source optimization tools for solving Capacitated Vehicle Route Problem with Time Windows, that is frequently used in practice. VROOM, OR-Tools and jsprit were compared on 56 Solomon benchmark problem instances. The obtained results have revealed some interesting properties of the considered optimization toolkits. VROOM turned out to be the most balances one in terms of solutions quality and execution time, while jsprit provides the best solutions in general. In many cases OR-Tools was also able to provide good quality results, however, for the most demanding instances was far from the two other considered tools. On the other hand, it has the most options that can be configured concerning the generation of an initial solution and the main heuristic. It can also be used in many environments (Python, C ++, Java, C #), while other tools are limited to only one.

REFERENCES

[1] LENSTRA, J.K., RINNOOY-KAN, A.H.G. Complexity of vehicle routing and scheduling problems. *Networks*. 1981. vol. 11, iss. 2, pp. 221-227.

[2] OROZCO, J. A microscopic traffic simulation based decision support system for real-time fleet management. Ph.D. Dissertation. 2011. Universitat Politecnica de Catalunya.

[3] CACERES-CRUZ, J., ARIAS, P., GUIMARANS, D., RIERA, D. and JUAN A.A. Rich Vehicle Routing Problem: Survey. ACM Computing Surveys. 2014. vol. 47, pp. 1-28.

[4] DANTZIG, G., RAMSER, J. The Truck Dispatching Problem. *Management Science*. 1959. vol. 6, pp. 80-91. http://dx.doi.org/10.1287/mnsc.6.1.80

[5] SAVELSBERGH, M.W.P. An efficient implementation of local search algorithms for constrained routing problems. *European Journal of Operational Research*. 1990. vol. 47, pp. 75-85.

[6] SOLOMON, M. M. Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research*. 1987. vol. 35, no. 2, pp. 254-265.

[7] HOMBERGER, J. and GEHRING, H. A two-phase hybrid metaheuristic for the vehicle routing problem with time windows. *European Journal of Operational Research*. 2005. vol. 162, pp. 220-238.

[8] BRÄYSY, O. and GENDREAU, M. Vehicle Routing Problem with Time Windows, Part II: Metaheuristics. *Transportation Science*. 2005. vol. 39, pp. 119-139.

[9] SHAW, P., Using constraint programming and local search methods to solve vehicle routing problems. *Lecture Notes in Computer Science*, Maher M., Puget J.F. (Eds.). 1998. vol. 1520, pp. 417-431.

[10] SCHRIMPF, G., SCHNEIDER, J., STAMM-WILBRANDT, H. and DUECK, G. Record Breaking Optimization Results Using the Ruin and Recreate Principle. *Journal of Computational Physics*. 2000. vol. 159, iss. 2, pp. 139-171