# Exploration on metaheuristics for solving VRPs

**CSI 5137 A Wei Li, 300113733**

## Abstract

Capacitated vehicle routing problem (CVRP) is an extension to the classic NP problem Travel salesman problem and can be solved by either exact method, heuristics methods or metaheurisitcs methods. We explore the nature of the CVRP and conduct a literature review focusing on the rich family of metaheuristics as well as how they are adapted to CVRP. Finally, we implemented an Ant Colony solver and compared it with Google OR-Tools and 3 other heuristics solvers.

## 1. Introduction

Vehicle routing problems is an extension and generalization to the TSP problem. In TSP, given an agent starting from an arbitrary node, we want to find a way it traverses all the nodes and return to the node with the minimal cost. Whereas in VRP (specifically, capacitated VRP), a depot is defined, the agent is assigned a number of loads and all nodes except the depot are assigned a number of demands. The agent needs to visit all the nodes to meet their demands, and when its loads are run out, it needs to go back to the depot for picking up additional loads. Specifically, we consider the simplest variation of VRP, the CVRP, which only considers capacity constraint. Many real-life applications can be formalized as VRP, such as goods logistics, parcel/food delivery. The problems add more constraint to the TPSs and are known to be NP-Hard, that is, it is at least as hard as NP, where no polynomial-time algorithm to find the exact solution, is supposed to exist.

Vehicle routing problems can be solved exhaustedly with exact methods like divide and conquer. They can also be solved approximately with heuristics or metaheuristics algorithms. Because of their similarity, algorithms that have been succeed in solving TSP also have been adapted to solve VRPs.

Exact solutions of VRPs can generally be found by exact methods. Comparing with other methods, exact methods provide exact solutions to a problem but also takes a significantly longer time. The most successful method is branch-and-bound which used the divide and conquer strategy. (Laporte & Nobert, 1987) gives a complete and detailed analysis of the branch-and-bound algorithms.

Unlike exact methods, Metaheuristics methods perform a relatively limited exploration of the search space. Some of the metaheuristics that are applied to VRPs are Tabu Search (Glover, 1986) and Evolutionary Algorithms and Ant Colony Optimization (ACO) (Dorigo et al., 1996). Under each family, there are a variety of different variations and different implementations to improve their performance and execution time.

In the project, we explore how metaheuristics like Ant Colony or Evolutionary algorithms can be used to solve VRPs. The rest of the report is organized as follows. In section 2, we discuss the VRP and combinatorial optimization, and identify the research questions and goals in the project. In section 3, we review related works of traditional ways and metaheuristics to solve CVRP and answer some of the research questions. In section 4, we discuss the methods that are used in this project. In section 5, the experiment setup, experiment details and experimental results are presented. Finally, the project report is concluded in section 6.

## 2. Problem Statement

As the project works on solving the capacitated vehicle routing problem, in this section, we introduce the TSP and CVRP. Also, we discuss how these problems can be expressed as a Combinatorial Optimization (CO) problem. Finally, we limit the scope and the question we want to answer in the project.

### 2.1. Travelling Salesman Problem (TSP)

The question of TSP is: given a list of nodes (cities/customers…), starting from an arbitrary node, what is the shortest possible route that visits each one of the nodes exactly once and finally returns to the starting node? In other words, given $N$ nodes coordinates in a 2D Euclidean space, we want to find the optimal permutation over the nodes such that the sum of distances between all adjacent nodes in the permutation is minimized. The problem can also be viewed as an agent (a salesman, a vehicle, etc.) makes decisions to choose the next node to visit given its current position and the neighboring nodes' information. In symmetric TSP, it is assumed that the distances between any two nodes are the same from either direction.

### 2.2. Capacitated Vehicle Routing Problem (CVRP)

The simplest form of VRP is the capacitated vehicle routing problem (CVRP). In CVRP, nodes are divided into customer nodes and a depot. Each customer node has a certain amount of demand to be met. One capacitated vehicle must start from the depot and visit all customer nodes to deliver some items. Each time a customer is visited, loads of the vehicle will be decreased. The vehicle must return to the depot for picking up additional items when the loads run out. The objective of VRP is to optimize the overall distance cost of the vehicle.

CVRP can also be formulated as multiple vehicles starting at the depot at the same time, and all the vehicles only return to the depot once. The vehicles (a fleet) can have either homogeneous or heterogeneous capacity. We consider the homogeneous capacitated problem. When all the vehicles have the same capacity, the problem is essentially the same as the previous CVRP. Also, if a customer node can be visited more than once, the problem is called CVRP with Split Delivery (CVRPSD). CVRP is considered more computationally difficult than TSP even with only a few hundred customer nodes.

### 2.3. Graph Representation

TSP and VRP can be described as graph theory problems. For CVRP, we define the problem as the following graph problem.

Let $G = (V + v_0, A, C)$ be a complete graph, where $V$ is the vertex set and $|V| = n$. $v_i \in V, i = 1, 2 \ldots, n$ denotes customer. $v_0$ is a single vertex stands for the depot. $A$ is the edge set with a nonnegative cost $c_{ij} \in C$ associate to it. $c_{ij}$ denotes the travel cost (often time or distance) spent to go from vertex $i$ to $j$. In the Symmetry CVRP case, $c_{ij} = c_{ji}$.

For each vertex $v_i$, there is a known nonnegative demand $d_i$. A set of $K$ identical vehicles each with capacity $B$ is are available in the depot (assuming vehicles are homogeneous). Also we assume that $d_i < B$ for all vertex and $\sum_{i=1}^{n} d_i \leq K * B$ so that the problem is feasible.

The problem of CVRP is defined as the following: finding at most $K$ circuits $S_i, i = 1, \ldots, K$, which is a permutation of vertex index, such that:

- each circuit visits the depot $v_0$
- each node $v_i$ is visited by exactly one circuit
- the sum of demand in a circuit does not exceed the vehicle capacity $B$: $\sum_{i \in S_i} d_i \leq B$
- the total cost of all circuit is minimized

### 2.4. Combinatorial Optimization (CO)

Both TSP and CVRP (and all of their variants) can also be viewed as a Combinatorial Optimization (CO) with different constraints. Combinatorial Optimization searches in a set instead of a real number domain and the objective is typically an integer, a subset, a permutation or a graph.

**Definition 1.1** A Combinatorial Optimization problem $P = (\mathcal{S}, f)$ can be defined by:

− a set of variables $X = \{x_1, \ldots, x_n\}$

− variable domains $D_1, \ldots, D_n$

− constraints among variables;

− an objective function $f$ to be minimized, where $f : D_1 \times \cdots \times D_n \rightarrow R^+$

The set of all possible feasible assignments is $\mathcal{S} = \{s = \{(x_1, v_1), \ldots, (x_n, v_n)\} | v_i \in D_i \; s$ satisfics all the constraints$\}$.

The goal of CO is to find the globally optimal solution: $s^*$: $f(s^*) \leq f(s), \forall s \in S$

To solve CO, we are interesting in the neighborhood structure of a solution, the neighborhood structure is defined as the following:

**Definition 1.2.** A neighborhood structure is a function $\mathcal{N} : \mathcal{S} \rightarrow 2^S$ that assigns to every $s \in \mathcal{S}$ a set of neighbors $\mathcal{N}(s) \subseteq \mathcal{S}$. $\mathcal{N}(s)$ is a subset of $S$ and is called the neighborhood of $s$.

With neighborhood structure, we define the local minimal solution as

**Definition 1.3.** A locally minimal solution (or local minimum) with respect to a neighborhood structure $\mathcal{N}$ is a solution $\hat{s}$ such that $\forall s \in \mathcal{N}(\hat{s}) : f(\hat{s}) \leq f(s)$.

With all that are defined, TSPs or VRPs fit well in the CO framework. TSP and VRP and their variations share many features: similar variable domain (a graph structure); similar solution structure (a permutation of nodes index); similar objective function (related to total distances, time, number of vehicles...) and similar neighborhood structure (eg. 2-opt). Consequently, these problems are closely related in terms of their CO representations.

### 2.5. Metaheuristics

A metaheuristic is an iterative master process that utilizes an underlying, more problem-specific heuristic iteration to efficiently produce high-quality optimization solutions.

Generally, closely following CO definition, all heuristics algorithms contain several components that are invariant to algorithms or problems. These components are: 1)representation: a representation to the structure of a valid solution, which is the $s$ in CO; 2) constraints defined by the problem; 3) Fitness function that decides the quality of a solution, which correspond to $f(s)$ in CO; 4) Operator used to generate new solutions in a neighborhood structure $\mathcal{N}(s) \subseteq \mathcal{S}$.

When applying a heuristics algorithm to solve an optimization problem, the algorithm needs specifically designed to accommodate the specific optimization problems. In other word, metaheuristics are not problem-specific. We want to explore the way metaheuristics algorithms being adapted to CVRP.

### 2.6. The Research Problem

In the project, we interest in the following research question: 1)How are the metaheuristics adapted to solve VRPs

- a. When solving TSP in the assignment, we used 2-OPT operation to modify solutions for local search, does the strategy still valid in VRP scenario?
- b. Can we use n-OPT instead of 2-OPT
- c. Are there other ways to build solutions for CVRP?

2) What kind of metaheuristics are used to solve VRPs

3) performance of search algorithms

- a. Time of execution
- b. Quality of solution, how close to the exact solution, if exist
- c. speed of convergence, converge rate

4) How are these metaheuristics implemented

In the next section, we talk about the family of metaheuristic methods to CVRP in detail and related these methods to the research questions.

## 3. Literature Review

We discuss briefly methods to solve CVRP and focus on the metaheuristics and methods related to the research problem.

### 3.1. Exact Methods

Methods that find the exact optimum exist. These approaches compute every possible solution until one of the bests is reached.

The most successful exact method comes from the branch-and-bound algorithm family. (Laporte & Nobert, 1987) gave a complete and detailed analysis of the branch-and-bound algorithms proposed up until the late 1980s. The branch-and-bound algorithm comes from the divide-and-conquers strategy, the basic idea is dividing CVRP into subproblems (which can be seen as a branch in a tree) and process subproblems in the queue order. The algorithm maintains a global best feasible solution as the upper bound to the problem. For each of the subproblem, it solves exactly some relaxed version of the problem, and use it as lower bound to the problem. When the lower bound matches the upper bound or all the subproblem is processed or pruned, the optimal is found, otherwise the algorithm branch and pruned the tree further.

Until the late 1980s, the most effective exact approaches for the CVRP were branch-and-bound algorithms which used basic combinatorial relaxations. The most recent branch-and-bound algorithms extend the algorithm to larger VRP size by using more advanced relaxation like Lagrangian relaxation(Fisher, 1994) or Additive Bounds(Fischetti et al., 1994).

Other successful exact method includes Set-Covering-Based and Branch-and-Cut Algorithms, which are widely used in TSP but are less researched for VRP.

Although there are exact methods, near all the methods that are used in practice are heuristics and metaheuristics because no exact algorithm guarantees to find the optimal solution within reasonable computing time when the number of nodes is large. This is due to the NP-Hardness of the problem.

### 3.2. Heuristics

Heuristics refers to optimization methods proposed to accommodate to VRP problem which only performs a relatively limited exploration of the search space. These methods typically produce good quality solutions within modest computing times. Unlike metaheuristics, these ”classic heuristics” algorithms are not general purposed. They are designed for VRP mostly between 1960 and 1990, more algorithmic, and generally does not include stochastic elements in the search. As a result, the exploration is not as deep as metaheuristics. Still, these methods are widely used in commercial packages

Heuristics can be classified into three categories, they are constructive heuristics, two-phase heuristics, and improvement methods. Constructive heuristics build a feasible solution ground up, adding nodes to a solution until it covers all the nodes, one of the most used constructive heuristics is Clarke Wright savings algorithm (Clarke & Wright, 1964). In two-phase heuristics, there are two stages, cluster stage cluster nodes into feasible routes, route stage build solution from clusters. The algorithm loops and feedback from the two phases. Finally, improvement methods attempt to upgrade a complete and feasible solution by exchanging vertex or edges in the solution.

A note to the heuristics is that the 2-opt technique we used in solving TSP is a type of improvement heuristics. The 2-opt is in fact a case of Lin's $\lambda$-opt mechanism(Lin, 1965), where $\lambda$ edges are removed from the routes and reconnected in all possible ways. Typically, the $\lambda$ is set to 2 or 3, (Renaud et al., 1996) developed a restricted version of 4-opt. As a generalization to the $\lambda$-opt, the Lin-Kernighan algorithm (Lin & Kernighan, 1973) is adaptive and at each step decides how many paths to be switched. (Helsgaun, 2000) implemented the Lin-Kernighan algorithm as the LKH framework with a larger family of the solver to solve TSP, VRP and their variation.

Finally, since most of the metaheuristics algorithms rely on local search to explore the neighborhood structure, they also heavily rely on improvement methods heuristics. However, there are also other ways to build solutions for CVRP, like the construction algorithm or two-phase algorithm, as discussed in 3.3.

### 3.3. Metaheuristics

Metaheuristics are general procedures that explore the solution space to identify good solutions, as described in section 2.5. Different metaheuristics, based on their characteristics, is adapted to embed some of the classic heuristics ideas about route construction and improvement for VRP. Metaheuristics algorithms that have been used for VRP include but not limited to: Simulated Annealing (SA), Tabu Search (TS), Genetic Algorithms (GA), Ant Systems (AS) and Neural Network (NN). The first three algorithms rely on constructing neighborhood structure for a complete solution, thus they can be seen as an extension to the improvement heuristics. On the other hand, AS and NN are constructive heuristics: new elements to the solution are added to the output as new information is gathered in each step of the iteration. During the process, Ant Systems relies on a dedicated probability model whereas Neural Network is a learnable parameterized model, often an encoder-decoder structure.

The early implementation of SA to VRP are (Robusté et al., 1990) where neighborhood structure is a combination of changing parts in a route and trading vertices between routes; (Appelgren, 1971) where the two-phase heuristics is used. (Osman, 1993) implemented SA in a more involved manner and achieved a more successful result. It uses the Clarke and Wright algorithm (Clarke & Wright, 1964) to obtain an initial solution, and perform exhausted local search using $\lambda$-opt neighborhood structure. It also implemented a more sophisticated cooling schedule.

Tabu search (TS) has been more extensively researched for VRP. In Tabu search, the solution space is explored by moving at each iteration from a solution $s$ to solutions in its neighborhood $\mathcal{N}(s)$, just like SA. Contrary to other metaheuristics, in TS, to avoid cycling, solutions that were recently examined are forbidden for a number of iterations. The duration that solutions remain tabu is called tabu-tenure. The tabu status can be overridden if certain conditions are met, for example, when a tabu solution is better than any previously seen solution. The basic concept of TS as described by (Glover & Laguna, 1993). The use of TS in VRP includes: Osman (Osman, 1993), where neighborhoods are defined by $\lambda$-opt; Taburoute(Gendreau et al., 1992), where neighborhoods are all solutions generated from removing a vertex from the current route and inserting it into another route containing one of its $p$ nearest neighbors using the Generalized Insertion procedure(Gendreau et al., 1992); (Xu & Kelly, 1996) whose neighborhoods are more involved, considering swaps of vertices between two routes, a global repositioning of some vertices into other routes, and local route improvements; Rochat and Taillard (Rochat & Taillard, 1995) which used the concept of adaptive memory procedure; Granular Tabu Search (Toth & Vigo, 2003) whose idea is eliminating all long edges whose length exceeds a granularity threshold. The intuition behind the operation is that longer edges of a graph have only a small likelihood of belonging to an optimal solution.

Although there are many applications of GA to TSP or VRP with more complex constraints, there are few adaptations of GA to the CVRP. To use GA for VRP, A member of the GA solutions population is typically represented as a string of index, where different routes are separated by the depot indices 0. For example, "0102304" indicates there are three routes, one has node 1, one has node 2 and 3, one has node 4. The work done on the CVRP was mostly aimed at evaluating the impact of different components or parameters of a GA. (Van Breedam, 1996) compares a GA with previously developed SA and TS. Best solutions produced by the three algorithms are of comparable quality, however, GA requires more computation time.

Ant system (AS) is proposed by (Colorni et al., 1992) who are inspired by the social behavior of ants. The ants' action is controlled by the goal of searching for food. While moving, ants deposit an organic compound called pheromone. Ants communicate with each other via pheromone trails, the more ants take the route the more pheromones are deposed, whereas in less visited routes, pheromones evaporate gradually. The first ant system for CVRP has been designed very recently by (Bullnheimer et al., 1999). In its design, there are two basic ant system phases: construction of vehicle routes and trail update. (Gambardella et al., 1999) has developed a multiple ant colony system for VRPTW, which is organized with a hierarchy of artificial ant colonies designed to successively optimize multiple objective functions.

Neural Network is among the most recent trend of researches that apply NN models to the field of Combinatorial Optimization. The NN models often rely on the constructive heuristic and encoder-decoder structures. Given a CVRP problem, the encoder encodes the information of the problem and current state to the vector space, and use another parametrized decoder to output the next node to be added to the existing solution. These models often view the process of building solutions as an interactive environment and use reinforcement learning to train the models.

## 4. Solution

In this section, we discuss the implementation part of the project. We designed and implemented a basic ant colony solver, and implemented the Google OR-Tools solver for CVRP, we also used the VeRyPy API to call three classic heuristics: parallel saving algorithm, sweep algorithm and two-phase heuristic. We apply these solvers to the set E problem in CVRPLIB, (Christofides & Eilon, 1969). The optimized cost, evaluation time are compared.

### 4.1. Data set

The data set used in the experiment is 11 problem instances in the (Christofides & Eilon, 1969). They are all in CVRPLIB format, which is like TSPLIB. Also, all the chosen problem instances are in the 2D-Euclidean format, and the known optimal solutions to the problems are given, allowing us to compare the obtained solution to the known best. The size of the problems ranges from 22 nodes to 101 nodes.

### 4.2. Ant Colony

The implementation of Ant Colony is based on the paper (Bullnheimer et al., 1999). In the implementation, each ant corresponds to a route in the solution (staying in the depot is also considered as a route). To solve the VRP, the artificial ants construct vehicle routes by successively choosing the next cities to visit. The travel of an ant end when the next city leads to an infeasibility because of low capacity, and the ant will choose the depot and start a new tour (as another ant). Each ant chooses the next node based on the information of pheromone, concretely, it is randomly proportional to the amount of pheromone in the edge and the cost of the edge. Once all cities have been visited and all routes in a solution are established, the quality of the solution will be evaluated, and the pheromone trace will be updated. After the above processes are finished, the algorithm starts the next step of optimization until the solver runs out of the given number of iterations. The idea of the algorithm can also be found in the flowchart in Figure 1.

In the experiment, we set the maximum number of ants to 22 and fix all executions' number of iterations to 350.

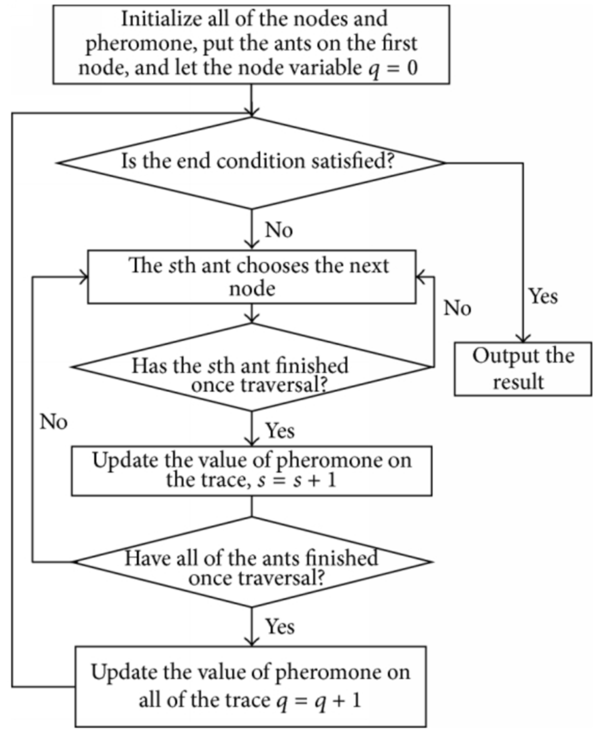


*Figure 1.* Flowchart of ACS

### 4.3. Google OR-Tools

OR-Tools is an open-source package for solving a range of operation research and optimization problems, including vehicle routing, flows, integer and linear programming and constraint programming. The package includes both tools to model complex problems with different constraints and many commercial or open-source solvers to solve the problems. Many works of literature about deep learning models for TSP and VRP use OR-Tools as a baseline model. In the project, OR-Tools is implemented to model CVRP. One thing of OR-Tools that is different from other solvers is that the stop criterion of OR-Tools is a specific running time instead of a number of iterations. We set the running time OR-Tools as 2 seconds (2000 milliseconds) and set the maximum number of vehicles as 22 for all problems.

### 4.4. VeRyPy

VeRyPy is an easy-to-use Python library of Capacitated Vehicle Routing Problem (CVRP) algorithms. It provides implementation and API to 15+ CVRP heuristics methods. We chose the three heuristics in this implementation for our experiments. These are: Clarke Wright (1964) parallel savings algorithm (Clarke & Wright, 1964), which is a classic and frequently used constructive heuristics; Sweep algorithm (Gillett & Miller, 1974), a cluster-first, routesecond heuristic; two-phase algorithm (Christofides, Mingozzi  Toth, 1979).

# 5. Results and Discussion

This section includes the setup of experiments, experimental results and discussion of the result.

## 5.1. Experiment Setup

The optimization experiment is run in the following setting: Ubuntu 18.04 subsystem in an Intel core i7 9700 + 16GB RAM machine. All implementation is written in Python.

## 5.2. Results

The five solvers, Ant Colony (aco), OR-Tools (ortools), Parallel Saving (savings), Sweep algorithm (sweep) and two-phase algorithm (2phase), were run against 11 problem instances in the Eilson data set. Data instances are identified in the format as 'n22-k4', the 'n22' stands for the number of nodes in the problem, including the depot; the 'k4' stands for the minimal number of vehicles used in the solution. However, we do not consider minimizing the number of vehicles in the experiment, the only objective is to minimize the distance traversed in the trips.

We record the optimization results value (fitness) and wall clock time during execution (exec time). Notes that the solvers do not share the same stopping criterion: 1) aco stops after 350 steps, 2) ortools stop after the given 2 seconds is run out 3)savings, sweep, 2phase are either constructive and deterministic or has the algorithm-specific stop criterion. As a result, the execution time comparison does not mean to be fair but only serves as a reference to execution efficiency. All time is recorded in milliseconds and taken the logarithm value.

The optimization result is given in Figure 2. The sweep algorithm is generally the worst performance one. Parallel saving is not as good in small problem size, but when problem size goes close to 100, its performance is among the best. ACO is the opposite of the Parallel Saving algorithm. Since all the problem instances are relatively small-size and easy, besides the sweep algorithm, all methods have similar performance close to the optimal.

The execution time is given in Figure 3. Our implementation of ACO takes an unreasonable long time for even the easiest problems, reflexing that the implementation is not very efficient. OR-Tools runs in the given 2 seconds. Finally, the three heuristics are very fast, maybe due to their deterministic nature and code optimization.

Finally, we compare the performance and execution time trade-off in Figure 4. The curve that is closer to the origin has a more desirable overall performance. It can be seen that the parallel saving algorithm is the best one in our experiment.

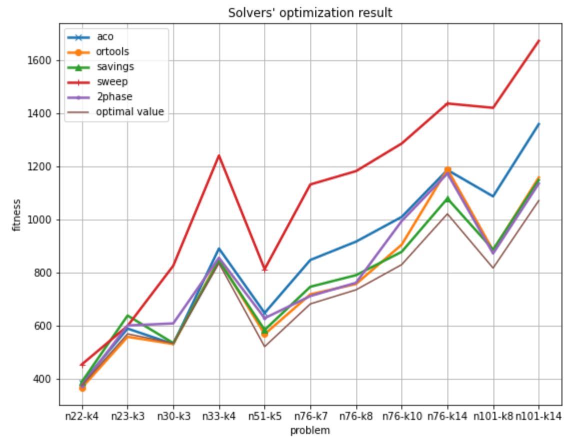


*Figure 2.* Solvers' optimization performance comparison

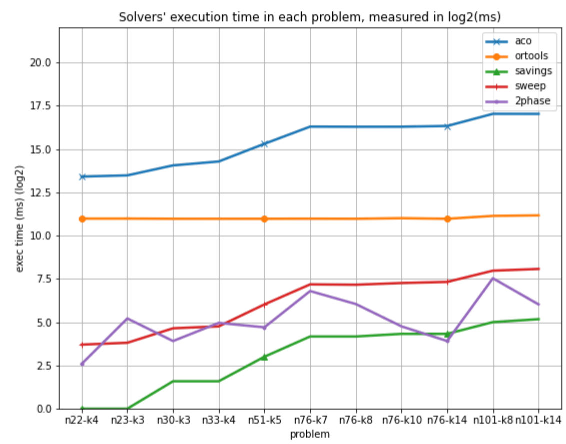


*Figure 3.* Solvers' execution time comparison

# 6. Conclusions

In the project, we explore the problem of capacitated vehicle routing and related methods to solve the problem. We conducted a literature review on the topic and focused specifically on the metaheuristics methods. Also, we implemented a naive ant colony solver, OR-Tools solver and a few classic heuristics solvers for CVRP, and compared their performance. The result shows that in simple problems, these solvers achieved similar performance. When execution time is considered, classic heuristics are better suited for small size problem.

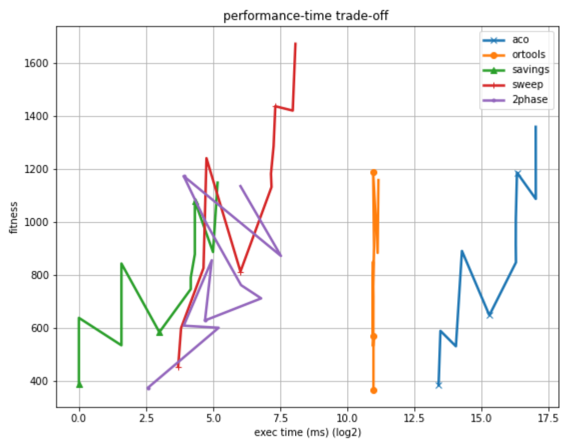


*Figure 4.* Solvers' time-performance trade-off comparison

## References


Appelgren, L. Integer programming methods for a vessel scheduling problem. *Transportation Science*, 5:64–78, 1971.

Bello, I., Pham, H., Le, Q. V., Norouzi, M., and Bengio, S. Neural combinatorial optimization with reinforcement learning. *ArXiv*, abs/1611.09940, 2017.

Bramel, J. and Simchi-Levi, D. On the effectiveness of set covering formulations for the vehicle routing problem with time windows. *Oper. Res.*, 45:295–301, 1997.

Bullnheimer, B., Hartl, R., and Strauss, C. Applying the ant system to the vehicle routing problem. 1999.

Christofides, N. and Eilon, S. An algorithm for the vehicle-dispatching problem. *Journal of the Operational Research Society*, 20:309–318, 1969.

Clarke, G. and Wright, J. W. Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research*, 12:568–581, 1964.

Colorni, A., Dorigo, M., Maniezzo, V., Varela, F., and Bourgine, P. Distributed optimization by ant colonies. 1992.

Cook, W. and Rich, J. A parallel cutting-plane algorithm for the vehicle routing problem with time windows. 1999.

Cordeau, J., Gendreau, M., and Laporte, G. A tabu search heuristic for periodic and multi-depot vehicle routing problems. *Networks*, 30:105–119, 1997.

Dorigo, M., Maniezzo, V., and Colorni, A. Ant system: optimization by a colony of cooperating agents. *IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics : a publication of the IEEE Systems, Man, and Cybernetics Society*, 26 1:29–41, 1996.

Falkner, J. and Schmidt-Thieme, L. Learning to solve vehicle routing problems with time windows through joint attention. *ArXiv*, abs/2006.09100, 2020.

Fischetti, M., Toth, P., and Vigo, D. A branch-and-bound algorithm for the capacitated vehicle routing problem on directed graphs. *Oper. Res.*, 42:846–859, 1994.

Fisher, M. Optimal solution of vehicle routing problems using minimum k-trees. *Oper. Res.*, 42:626–642, 1994.

Gambardella, L., Taillard, É., and Agazzi, G. Macs-vrptw: a multiple ant colony system for vehicle routing problems with time windows. 1999.

Gendreau, M., Hertz, A., and Laporte, G. New insertion and postoptimization procedures for the traveling salesman problem. *Oper. Res.*, 40:1086–1094, 1992.

Gendreau, M., Hertz, A., Laporte, G., and Stan, M. A generalized insertion heuristic for the traveling salesman problem with time windows. *Oper. Res.*, 46:330–335, 1998.

Gillett, B. and Miller, L. A heuristic algorithm for the vehicle-dispatch problem. *Oper. Res.*, 22:340–349, 1974.

Glover, F. and Laguna, M. In modern heuristic techniques for combinatorial problems. 1993.

Glover, F. W. Future paths for integer programming and links to artificial intelligence. *Comput. Oper. Res.*, 13: 533–549, 1986.

Helsgaun, K. An effective implementation of the lin-kernighan traveling salesman heuristic. *Eur. J. Oper. Res.*, 126:106–130, 2000.

Khalil, E. B., Dai, H., Zhang, Y., Dilkina, B., and Song, L. Learning combinatorial optimization algorithms over graphs. *ArXiv*, abs/1704.01665, 2017.

Knight, K. W. and Hofer, J. Vehicle scheduling with timed and connected calls: A case study. *Journal of the Operational Research Society*, 19:299–310, 1968.

Kool, W., Van Hoof, H., and Welling, M. Attention, learn to solve routing problems! *arXiv preprint arXiv:1803.08475*, 2018.

Laporte, G. and Nobert, Y. Exact algorithms for the vehicle routing problem. *North-holland Mathematics Studies*, 132:147–184, 1987.

Lin, S. Computer solutions of the traveling salesman problem. *Bell System Technical Journal*, 44:2245–2269, 1965.

Lin, S. and Kernighan, B. An effective heuristic algorithm for the traveling-salesman problem. *Oper. Res.*, 21:498–516, 1973.

Ma, Q., Ge, S., He, D., Thaker, D., and Drori, I. Combinatorial optimization by graph pointer networks and hierarchical reinforcement learning. *ArXiv*, abs/1911.04936, 2019.

Nazari, M., Oroojlooy, A., Snyder, L. V., and Takác, M. Reinforcement learning for solving the vehicle routing problem. In *NeurIPS*, 2018.

Osman, I. Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem. *Annals of Operations Research*, 41:421–451, 1993.

Potvin, J., Kervahut, T., Garcia, B.-L., and Rousseau, J. The vehicle routing problem with time windows part i: Tabu search. *INFORMS J. Comput.*, 8:158–164, 1996.

Renaud, J., Boctor, F., and Laporte, G. A fast composite heuristic for the symmetric traveling salesman problem. *INFORMS J. Comput.*, 8:134–143, 1996.

Robusté, F., Daganzo, C., and Souleyrette, R. Implementing vehicle routing models. *Transportation Research Part B-methodological*, 24:263–286, 1990.

Rochat, Y. and Taillard, É. Probabilistic diversification and intensification in local search for vehicle routing. *Journal of Heuristics*, 1:147–167, 1995.

Russell, R. Hybrid heuristics for the vehicle routing problem with time windows. *Transp. Sci.*, 29:156–166, 1995.

Toth, P. and Vigo, D. The granular tabu search and its application to the vehicle-routing problem. *INFORMS J. Comput.*, 15:333–346, 2003.

Uchoa, E., Pecin, D., Pessoa, A., Poggi, M., Vidal, T., and Subramanian, A. New benchmark instances for the capacitated vehicle routing problem. *European Journal of Operational Research*, 257(3):845 – 858, 2017. ISSN 0377-2217. doi: https://doi.org/10.1016/j.ejor.2016.08.012. URL http://www.sciencedirect.com/science/article/pii/S0377221716306270.

Van Breedam, A. An analysis of the effect of local improvement operators in genetic algorithms and simulated annealing for the vehicle routing problem. 01 1996.

Wu, Y., Song, W., Cao, Z., Zhang, J., and Lim, A. Learning improvement heuristics for solving routing problems. *arXiv: Artificial Intelligence*, 2019.

Xu, J. and Kelly, J. P. A network flow-based tabu search heuristic for the vehicle routing problem. *Transp. Sci.*, 30:379–393, 1996.