



Optimization Methods for Vehicle Routing in Transportation Planning and Logistics: A Structured Review and Algorithm-Selection Framework

T. Thi Lan^{1,*}

¹ Department of Logistics and Supply Chain Management, Faculty of Management, Vietnam National University, Ho Chi Minh City, Vietnam

ARTICLE INFO	ABSTRACT
Article History: Received 15 March 2025 Received in revised form 9 June 2025 Accepted 11 July 2025 Available online 20 September 2025	Vehicle routing lies at the core of transportation planning and logistics, and a large body of Operations Research (OR) methods has been developed to solve it. This paper presents a structured narrative review of optimization methods for the capacitated vehicle routing problem (CVRP) and its principal extensions. The review covers mixed-integer programming formulations, exact branch-and-cut and branch-and-price algorithms, classical construction heuristics, modern metaheuristics (tabu search, simulated annealing, genetic algorithms, ant colony optimization, particle swarm optimization, adaptive large neighborhood search and hybrid genetic search), matheuristics and learning-based approaches, using a compact mixed-integer linear programming (MILP) model as a common reference formulation. We propose a classification framework that organizes methods by solution guarantee, search paradigm and suitability for different problem scales, and derive practical guidelines for algorithm selection based on instance size, constraint structure, uncertainty and available computing time. We also summarize standards for fair computational comparison, as inconsistent benchmarking is a recurrent weakness of the applied metaheuristic literature. No single method dominates: exact methods remain indispensable for small or well-structured instances and for computing bounds, whereas well-engineered hybrid metaheuristics are preferred at scale, and several population-based algorithms popular in applied studies are generally less competitive than neighborhood-search-based methods on standard benchmarks. Open research directions include integrating machine learning with optimization, dynamic and stochastic routing, and reproducible benchmarking.
Keywords: Operations Research; Vehicle Routing Problem; Mixed-Integer Programming; Metaheuristics; Transportation Planning; Logistics	

1. INTRODUCTION

Transportation accounts for a large share of the cost of distributing goods, and decisions about which vehicle serves which customer, and in what order, directly determine cost, service quality, and emissions. Operations Research (OR) has supported these decisions since its early days. The truck dispatching problem was formulated as

* Corresponding Author: Tran.ThiLan@gmail.com

Department of Logistics and Supply Chain Management, Faculty of Management, Vietnam National University, Ho Chi Minh City, Vietnam



an optimization problem in 1959 [1], and the vehicle routing problem (VRP) has since become one of the most studied problems in combinatorial optimization [2,3].

The methods available to solve routing problems are diverse. Linear programming (LP) and network-flow models describe assignment and transportation decisions efficiently [4,5]; mixed-integer programming (MIP) captures the discrete structure of routes; and because the VRP is NP-hard [6], exact algorithms become impractical beyond moderate instance sizes, which has motivated a rich family of heuristics and metaheuristics [7,8]. For a practitioner or researcher, the practical question is therefore not whether such methods exist, but which of them is appropriate for a given routing problem.

Excellent surveys already exist, but they are typically organized by problem variant [9] or by algorithm family [3,8]. Applied studies, in contrast, frequently choose an algorithm for convenience, often a general-purpose nature-inspired method, and evaluate it on a few self-generated instances without strong baselines. This makes results hard to compare and can give a misleading impression of the relative merits of methods [10,11]. This paper addresses that gap from a decision-oriented perspective.

The contributions of this paper are as follows. First, a compact MILP formulation of the CVRP is presented as a common reference model, and its limitations and main extensions are discussed. Second, a classification framework is proposed that organizes exact, heuristic, metaheuristic, hybrid, and learning-based methods according to solution guarantee, search paradigm, and scale of applicability. Third, practical guidelines for algorithm selection and standards for computational experimentation are derived from the literature. No new computational experiments are reported; the conclusions rest on the cited literature and on the standard benchmark instances it employs.

The remainder of the paper is organized as follows. Section 2 summarizes the OR background. Section 3 formulates the CVRP. Section 4 reviews solution approaches. Section 5 presents the classification framework, Section 6 the selection guidelines, and Section 7 the recommended standards for computational comparison. Section 8 discusses research directions, and Section 9 concludes.

2. BACKGROUND: OPTIMIZATION IN TRANSPORTATION AND LOGISTICS

2.1. Linear and network models

Linear programming was developed in the late 1930s and 1940s, with early formulations of production and transportation planning by Kantorovich [12] and the introduction of the simplex method by Dantzig [4]. The classical transportation problem, which assigns shipments from supply points to demand points at minimum cost, is an LP with network structure; when supplies and demands are integers, its optimal basic solutions are integral, so it can be solved as a plain LP [5]. Interior-point methods, initiated by Karmarkar [13], complement the simplex method, and modern solvers rely on dual simplex and barrier algorithms [14]. These models are central to fleet allocation and freight flow planning, but they assume divisible, continuous decisions and cannot represent the sequencing of stops in a route.

2.2. Nonlinear and integer models

Nonlinear programming (NLP) is appropriate when costs or constraints are smooth nonlinear functions of continuous variables, for example fuel consumption as a function of speed or load. It is solved with gradient-based, Newton, and quasi-Newton methods [15]. Routing decisions, however, are inherently discrete: a customer is either visited by a vehicle or not, and one arc follows another or not. Routing problems are therefore modeled with integer or mixed-integer programs, which are solved by branch-and-bound and its enhancements such as cutting planes [14]. NLP methods thus play a supporting role in routing, for instance in speed optimization on a fixed route, rather than a central one.

2.3. Computational complexity

The traveling salesman problem (TSP) is a special case of the VRP, and both are NP-hard [6]. Unlike the transportation problem, the VRP has no known polynomial-time algorithm, and the worst-case effort of exact methods grows rapidly with the number of customers. This distinction between polynomially solvable network problems and NP-hard routing problems underlies the division of solution methods discussed in Section 4.

3. 3. PROBLEM FORMULATION

3.1. Problem description

The capacitated vehicle routing problem (CVRP) is the reference problem of this study. A fleet of at most K identical vehicles, each with capacity Q , is based at a central depot. A set of customers, each with a known demand, must be served. Each customer is visited exactly once, each route starts and ends at the depot, and the total demand on a route cannot exceed Q . The objective is to minimize the total travel cost (distance or time) [1,3].

3.2. Notation

Sets: $N = \{0, 1, \dots, n\}$ is the set of nodes, where 0 is the depot; $C = N \setminus \{0\}$ is the set of customers.

Parameters: c_{ij} is the travel cost from node i to node j ; d_i is the demand of customer i ; Q is the vehicle capacity; K is the maximum number of vehicles.

Variables: $x_{ij} \in \{0,1\}$ equals 1 if a vehicle travels directly from node i to node j ; u_i is the cumulative load on the vehicle after serving customer i .

3.3. Mixed-integer linear programming model

The model below is the standard compact two-index formulation, in which the subtour-elimination and capacity requirements are enforced through the Miller–Tucker–Zemlin (MTZ) constraints [16].

$$\min \sum_{i \in N} \sum_{j \in N, j \neq i} c_{ij} x_{ij} \quad (1)$$

subject to

$$\sum_{j \in N, j \neq i} x_{ij} = 1 \quad \forall i \in C \quad (2)$$

$$\sum_{i \in N, i \neq j} x_{ij} = 1 \quad \forall j \in C \quad (3)$$

$$\sum_{j \in C} x_{0j} = \sum_{i \in C} x_{i0} \leq K \quad (4)$$

$$u_j \geq u_i + d_j - Q(1 - x_{ij}) \quad \forall i, j \in C, i \neq j \quad (5)$$

$$d_i \leq u_i \leq Q \quad \forall i \in C \quad (6)$$

$$x_{ij} \in \{0,1\} \quad \forall i, j \in N, i \neq j \quad (7)$$

Equation (1) minimizes total travel cost. Constraints (2) and (3) require that every customer has exactly one successor and one predecessor. Constraint (4) states that as many vehicles leave the depot as return to it, and that at most K are used. Constraints (5) and (6) track the cumulative load along each route: they forbid subtours that do not contain the depot and guarantee that no route exceeds the capacity Q . Constraint (7) defines the binary routing variables.

3.4. Classification and remarks

The model contains binary variables x_{ij} and continuous variables u_i , and its objective and constraints are linear, so it is a mixed-integer linear program (MILP). It can be passed directly to general-purpose MIP solvers. Two remarks are important. First, the linear relaxation of MTZ-based formulations is weak, so branch-and-bound trees grow quickly and such models are practical only for small instances [3]. Second, formulations with exponentially many capacity and connectivity cuts, or set-partitioning formulations over routes, give much tighter bounds and are the basis of state-of-the-art exact algorithms [17,18].

3.5. Extensions

Practical logistics problems add features to the CVRP. Time windows lead to the VRP with time windows (VRPTW), which requires arrival-time variables and time-consistency constraints [19]; heterogeneous fleets require vehicle-dependent capacities and costs (a vehicle index on the routing variables); other common variants include multiple depots, pickup and delivery, split deliveries, and periodic routing [3,8]. Uncertainty in demands or travel times and the need for real-time re-optimization lead to stochastic and dynamic variants [20,21]. Each extension changes which solution methods remain effective, which motivates the classification in Section 5.

4. SOLUTION APPROACHES

4.1. Exact methods

Exact methods return a provably optimal solution, or a solution together with a proven bound on its distance from the optimum. Compact MILP formulations such as the model in Section 3.3 are easy to implement and solve with commercial or open-source solvers, but scale poorly. More effective approaches strengthen the formulation with valid inequalities separated dynamically (branch-and-cut) or reformulate the problem over an exponential set of routes and generate them on demand (column generation and branch-and-price). Branch-cut-and-price algorithms combine both ideas and currently solve CVRP instances with up to a few hundred customers to optimality [17,18].

Two points are essential when exact methods are compared with heuristics. When optimality is proven, the exact solution cannot be worse than that of a heuristic on the same model and data. If a solver is stopped by a time limit, it should be reported as such, with the final optimality gap, rather than as an optimal result. Exact methods are also valuable even at large scale because their bounds allow the quality of heuristic solutions to be assessed.

4.2. Classical heuristics

Classical construction heuristics build routes directly. The savings algorithm merges routes in decreasing order of the cost saved by joining them [22], and the sweep algorithm assigns customers to vehicles by polar angle around the depot [23]. These are fast and simple and are typically combined with local-search improvement moves such as relocating or exchanging customers and 2-opt on routes [7]. Their solutions are usually of moderate quality, so their main roles are as baselines and as initial solutions for metaheuristics.

4.3. Metaheuristics

Metaheuristics are general search frameworks that guide subordinate heuristics beyond the first local optimum, without guaranteeing global optimality [24]. They are widely used for VRP because they handle large instances and additional constraints flexibly. We distinguish local-search-based and population-based frameworks.

4.3.1. Local-search-based methods

Tabu search accepts non-improving moves while forbidding recently visited solution attributes, and was among the first metaheuristics to achieve strong results on the VRP [25]. Simulated annealing accepts worse solutions with a probability that decreases over time [26]. Adaptive large neighborhood search (ALNS) repeatedly destroys and repairs part of the solution using a portfolio of operators whose selection probabilities are adapted based on past success [27]; it is popular because new constraints can often be handled by adding operators, and it performs well across many routing variants [8].

4.3.2. Population-based methods

Genetic algorithms evolve a population of solutions using selection, crossover, and mutation [28]. For routing, the encoding is critical: a common and effective approach represents a solution as a giant tour that is split into feasible routes and improved by local search, giving memetic algorithms [29]. Hybrid genetic search (HGS) extends this idea with diversity management and penalized handling of infeasible solutions, and its recent implementation is among the strongest heuristics for the CVRP on standard benchmarks [30,31]. A plain genetic algorithm without local search is, by contrast, generally not competitive.

Ant colony optimization builds solutions incrementally using pheromone trails and heuristic desirability [32], and has been adapted to the VRP [33]. Particle swarm optimization was designed for continuous search spaces [34]; applying it to routing requires a mapping from continuous particle positions to permutations and routes, and it is generally combined with local search to be effective. Both methods are conceptually simple and appear frequently in applied studies. However, on standard CVRP benchmarks, they are generally reported to be less competitive than

tabu search, ALNS, or HGS unless strongly hybridized [8,31]. It has also been argued that the metaphor of a method should not substitute for a clear description of its search components and a rigorous evaluation [11].

A recent scalable heuristic shows that carefully designed local search with problem-specific data structures can handle instances with many thousands of customers [35], illustrating that scalability depends more on the design of the neighborhood and data structures than on the metaheuristic family.

4.4. Matheuristics and hybrid methods

Matheuristics combine mathematical programming with heuristic search, for example by solving a set-partitioning problem over a pool of routes generated by a metaheuristic, or by using a MIP solver to explore large neighborhoods. Such hybrids seek to combine the flexibility and speed of heuristics with the bounding and optimization strength of exact methods, and hybridization is a central theme in modern metaheuristics research [24].

4.5. Uncertainty, dynamics, and learning-based methods

When demands or travel times are uncertain, stochastic and robust formulations and simulation-based evaluation of solutions are used [20]; when requests arrive during execution, dynamic routing methods re-optimize as information becomes available [21]. Machine learning has recently been applied to combinatorial optimization, either to learn constructive policies end-to-end or to improve components of classical algorithms such as branching or operator selection [36]. Reinforcement-learning and attention-based models for routing can generate solutions very quickly once trained [37,38], but they generally do not yet match the solution quality of state-of-the-art heuristics on large standard instances, and their generalization to different instance distributions remains a concern [36]. Learning is therefore currently most promising as a component of optimization algorithms, or for settings in which many similar instances must be solved very quickly.

5. CLASSIFICATION FRAMEWORK

We organize solution methods along three dimensions. The first is the solution guarantee: whether the method can prove optimality or bound the gap (exact) or provides no guarantee (heuristic). The second is the search paradigm: mathematical programming, construction, local search, population-based search, hybrid, or learning-based. The third is the scale and constraint richness at which the method is typically effective. Table 1 summarizes the resulting classes; the statements on scale are indicative orders of magnitude drawn from the cited literature, not results of this study.

Table 1. Classification of solution methods for vehicle routing problems

Family	Representative methods	Optimality guarantee	Typical scale	Main strengths	Main limitations
Compact MIP	MTZ-based MILP with a general-purpose solver	Yes, if gap is closed	Small (tens of customers)	Easy to implement; flexible; provides bounds	Weak relaxation; poor scalability
Advanced exact	Branch-and-cut, branch-and-price, branch-cut-and-price [17,18]	Yes	Up to a few hundred customers (CVRP)	Proven optimality; strong bounds	Complex and problem-specific; limited scale
Classical heuristics	Savings, sweep, local-search moves [22,23]	No	Very large	Fast and simple; useful baselines	Moderate quality; limited handling of complex constraints
Local-search metaheuristics	Tabu search, simulated annealing, ALNS [25,26,27]	No	Medium to very large	High quality; flexible constraint handling	Requires operator and parameter design
Population-based hybrids	Memetic algorithms, HGS [29,30,31]	No	Medium to large	State-of-the-art quality on	Implementation complexity

				standard CVRP benchmarks	
Swarm-inspired methods	Basic PSO and ACO [32,34]	No	Small to medium in most reported studies	Conceptually simple; ACO suits constructive search	Generally less competitive without strong local search; encoding required
Matheuristics	Route-pool set partitioning; MIP-based neighborhoods	Usually no	Medium to large	Combine bounding and heuristic search	Need a MIP solver; tuning effort
Learning-based	RL and attention-based constructive policies [37,38]	No	Fast inference at scale	Speed after training; adapt to an instance distribution	Training cost; generalization; quality gap

Two observations follow from the framework. First, the ordering by guarantee is the reverse of the ordering by scalability, which explains why the methods coexist rather than replace one another. Second, within the metaheuristic class, differences in performance are driven mainly by the quality of the local-search components, the handling of infeasibility, and diversity management, not by the source of the metaphor [11,31].

6. GUIDELINES FOR ALGORITHM SELECTION

Table 2 translates the framework into practical recommendations. They should be read as starting points: the final choice should be validated on representative instances of the actual application.

Table 2. Guidelines for selecting a solution approach

Situation	Suggested approach	Rationale
Small instance (tens of customers); proof of optimality required	Compact MILP or branch-and-cut with a MIP solver	Optimality is provable at low development cost
Medium instance; standard CVRP or VRPTW; high quality required and hours available	Advanced exact method if available; otherwise a state-of-the-art hybrid metaheuristic together with bounds	Best attainable quality, with an assessment of the gap
Large instance (hundreds to thousands of customers) and limited time	ALNS, HGS-type, or scalable local search	Good solutions in short time [31,35]
Rich constraints (time windows, heterogeneous fleet, pickup and delivery)	ALNS or a general metaheuristic with penalized infeasibility	Constraints are added through operators and penalties [8,27]
Real-time or dynamic re-optimization	Fast construction plus local search from the current solution; learned policies if many similar instances exist	Latency is the main criterion [21]
Uncertain demands or travel times	Stochastic or robust formulations; simulation-based evaluation of candidate solutions	Deterministic optimality may not be robust [20]
A quick baseline is needed	Savings algorithm followed by local search	Simple and reproducible [7,22]

Three general principles apply. First, one should start with the problem structure: the presence of a strong exact formulation, or of a decomposition that exposes a network or set-partitioning structure, may change the recommendation. Second, a simple, strong baseline should be established before more elaborate methods are considered. Third, when a swarm-inspired or other general-purpose algorithm is proposed for a standard routing problem, evidence of its competitiveness against the methods in Table 1 is required.

7. STANDARDS FOR COMPUTATIONAL COMPARISON

Many of the conclusions drawn in applied routing studies are unreliable because of weaknesses in experimental design [10,39]. On the basis of the literature, we recommend the following minimum standards for studies comparing routing algorithms.

- Use standard benchmark instances, such as Solomon instances for time windows [19] and the instance set of Uchoa et al. for the CVRP [40], so that results are comparable with the literature; when instances are generated, describe the generator and provide the data.
- Include strong baselines that reflect the state of the art, not only similar algorithms of the same family [31].
- Report exact methods with the time limit, the best bound, and the final optimality gap; do not describe a time-limited run as optimal.
- For stochastic algorithms, run several independent replications with recorded random seeds, and report the best, mean, worst, and standard deviation of the objective. Test differences with an appropriate non-parametric statistical test before ranking methods.
- Use equal and stated computing budgets, and document hardware, software versions, and programming language. Tune parameters systematically with a documented procedure such as iterated racing [41], using instances separate from those used for evaluation.
- Report solution quality and computing time together; time alone says little about the value of a method. Define any convergence or robustness measure operationally.
- Make the source code available so that the results can be reproduced.

8. RESEARCH GAPS AND FUTURE DIRECTIONS

- Integration of machine learning and optimization. Learning can select operators, configure parameters, predict promising arcs, or provide warm starts; the open challenge is to show consistent gains over strong, well-tuned baselines and to generalize across instance distributions [36].
- Dynamic and stochastic routing. Many real systems operate under uncertain demand and travel times and require decisions in real time; combining simulation, stochastic optimization, and fast re-optimization remains an active area [20,21].
- Hybrid and matheuristic algorithms. Combining the bounds of exact methods with the scalability of metaheuristics is a promising route to solving richer problems with quality guarantees [24].
- Parallel and distributed computation. Exploiting modern hardware for population-based and decomposition approaches is relevant to very large-scale and time-critical applications.
- Rigorous and reproducible benchmarking. Wider adoption of the standards in Section 7 and of shared benchmark libraries would make it easier to identify methods that genuinely advance the state of the art [11,40].

9. CONCLUSION

This paper reviewed optimization methods for vehicle routing in transportation planning and logistics, using the CVRP as the reference problem. A compact MILP formulation was presented, methods from exact algorithms to learning-based approaches were organized in a classification framework, and guidelines for algorithm selection and for computational comparison were derived from the literature.

The main conclusions are as follows. No method dominates across all situations: exact methods provide optimality proofs and bounds for small or structured problems, whereas hybrid metaheuristics based on strong local search, such as ALNS and HGS, are the appropriate choice for medium and large instances. Nature-inspired

algorithms such as basic PSO and ACO are conceptually attractive but generally require substantial hybridization to be competitive on standard benchmarks. Finally, the credibility of any comparison depends on standard benchmarks, strong baselines, statistical rigor, and full reporting.

This study has limitations. It is a narrative review focused on the CVRP and its main variants and does not report new experiments. Statements about relative performance reflect the cited literature and may change as new algorithms appear. Future work could complement the framework with a systematic, benchmark-based experimental comparison that follows the standards proposed in Section 7.

Authors' Contributions

All authors contributed to the conception and design of the study, the interpretation of the results and the preparation of the manuscript. All authors read and approved the final manuscript.

Declaration of Generative AI Use

During the preparation of this work, the authors used generative AI tools to improve the language and readability of the manuscript. After using these tools, the authors reviewed and edited the content as needed and take full responsibility for the content of the publication.

Acknowledgments

The authors are grateful to all those who contributed to this project.

Declaration of Competing Interests

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Funding

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Ethics Considerations

Not applicable. This article is a review of previously published literature and did not involve human participants, animals or personal data.

Data Availability

No new data were generated or analyzed in this study.

REFERENCES

- [1] Dantzig, G. B., & Ramser, J. H. (1959). The truck dispatching problem. *Management Science*, 6(1), 80–91. <https://doi.org/10.1287/mnsc.6.1.80>
- [2] Laporte, G. (2009). Fifty years of vehicle routing. *Transportation Science*, 43(4), 408–416. <https://doi.org/10.1287/trsc.1090.0301>
- [3] Toth, P., & Vigo, D. (Eds.). (2014). *Vehicle routing: Problems, methods, and applications* (2nd ed.). SIAM. <https://doi.org/10.1137/1.9781611973594>

- [4] Dantzig, G. B. (1963). *Linear programming and extensions*. Princeton University Press. <https://doi.org/10.1515/9781400884179>
- [5] Ahuja, R. K., Magnanti, T. L., & Orlin, J. B. (1993). *Network flows: Theory, algorithms, and applications*. Prentice Hall.
- [6] Lenstra, J. K., & Rinnooy Kan, A. H. G. (1981). Complexity of vehicle routing and scheduling problems. *Networks*, 11(2), 221–227. <https://doi.org/10.1002/net.3230110211>
- [7] Cordeau, J.-F., Gendreau, M., Laporte, G., Potvin, J.-Y., & Semet, F. (2002). A guide to vehicle routing heuristics. *Journal of the Operational Research Society*, 53(5), 512–522. <https://doi.org/10.1057/palgrave.jors.2601319>
- [8] Vidal, T., Crainic, T. G., Gendreau, M., & Prins, C. (2013). Heuristics for multi-attribute vehicle routing problems: A survey and synthesis. *European Journal of Operational Research*, 231(1), 1–21. <https://doi.org/10.1016/j.ejor.2013.02.053>
- [9] Braekers, K., Ramaekers, K., & Van Nieuwenhuysse, I. (2016). The vehicle routing problem: State of the art classification and review. *Computers & Industrial Engineering*, 99, 300–313. <https://doi.org/10.1016/j.cie.2015.12.007>
- [10] Hooker, J. N. (1995). Testing heuristics: We have it all wrong. *Journal of Heuristics*, 1(1), 33–42. <https://doi.org/10.1007/BF02430364>
- [11] Sörensen, K. (2015). Metaheuristics—the metaphor exposed. *International Transactions in Operational Research*, 22(1), 3–18. <https://doi.org/10.1111/itor.12001>
- [12] Kantorovich, L. V. (1960). Mathematical methods of organizing and planning production. *Management Science*, 6(4), 366–422. <https://doi.org/10.1287/mnsc.6.4.366>
- [13] Karmarkar, N. (1984). A new polynomial-time algorithm for linear programming. *Combinatorica*, 4(4), 373–395. <https://doi.org/10.1007/BF02579150>
- [14] Bixby, R. E. (2012). A brief history of linear and mixed-integer programming computation. *Documenta Mathematica, Extra Volume ISMP*, 107–121. <https://doi.org/10.4171/dms/6/16>
- [15] Nocedal, J., & Wright, S. J. (2006). *Numerical optimization* (2nd ed.). Springer. <https://doi.org/10.1007/978-0-387-40065-5>
- [16] Miller, C. E., Tucker, A. W., & Zemlin, R. A. (1960). Integer programming formulation of traveling salesman problems. *Journal of the ACM*, 7(4), 326–329. <https://doi.org/10.1145/321043.321046>
- [17] Baldacci, R., Mingozzi, A., & Roberti, R. (2012). Recent exact algorithms for solving the vehicle routing problem under capacity and time window constraints. *European Journal of Operational Research*, 218(1), 1–6. <https://doi.org/10.1016/j.ejor.2011.07.037>
- [18] Pecin, D., Pessoa, A., Poggi, M., & Uchoa, E. (2017). Improved branch-cut-and-price for capacitated vehicle routing. *Mathematical Programming Computation*, 9(1), 61–100. <https://doi.org/10.1007/s12532-016-0108-8>
- [19] Solomon, M. M. (1987). Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research*, 35(2), 254–265. <https://doi.org/10.1287/opre.35.2.254>
- [20] Gendreau, M., Laporte, G., & Séguin, R. (1996). Stochastic vehicle routing. *European Journal of Operational Research*, 88(1), 3–12. [https://doi.org/10.1016/0377-2217\(95\)00050-X](https://doi.org/10.1016/0377-2217(95)00050-X)

- [21] Pillac, V., Gendreau, M., Guéret, C., & Medaglia, A. L. (2013). A review of dynamic vehicle routing problems. *European Journal of Operational Research*, 225(1), 1–11. <https://doi.org/10.1016/j.ejor.2012.08.015>
- [22] Clarke, G., & Wright, J. W. (1964). Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research*, 12(4), 568–581. <https://doi.org/10.1287/opre.12.4.568>
- [23] Gillett, B. E., & Miller, L. R. (1974). A heuristic algorithm for the vehicle-dispatch problem. *Operations Research*, 22(2), 340–349. <https://doi.org/10.1287/opre.22.2.340>
- [24] Blum, C., & Roli, A. (2003). Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Computing Surveys*, 35(3), 268–308. <https://doi.org/10.1145/937503.937505>
- [25] Gendreau, M., Hertz, A., & Laporte, G. (1994). A tabu search heuristic for the vehicle routing problem. *Management Science*, 40(10), 1276–1290. <https://doi.org/10.1287/mnsc.40.10.1276>
- [26] Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, 220(4598), 671–680. <https://doi.org/10.1126/science.220.4598.671>
- [27] Ropke, S., & Pisinger, D. (2006). An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation Science*, 40(4), 455–472. <https://doi.org/10.1287/trsc.1050.0135>
- [28] Holland, J. H. (1975). *Adaptation in natural and artificial systems*. University of Michigan Press.
- [29] Prins, C. (2004). A simple and effective evolutionary algorithm for the vehicle routing problem. *Computers & Operations Research*, 31(12), 1985–2002. [https://doi.org/10.1016/S0305-0548\(03\)00158-8](https://doi.org/10.1016/S0305-0548(03)00158-8)
- [30] Vidal, T., Crainic, T. G., Gendreau, M., Lahrichi, N., & Rei, W. (2012). A hybrid genetic algorithm for multidepot and periodic vehicle routing problems. *Operations Research*, 60(3), 611–624. <https://doi.org/10.1287/opre.1120.1048>
- [31] Vidal, T. (2022). Hybrid genetic search for the CVRP: Open-source implementation and SWAP* neighborhood. *Computers & Operations Research*, 140, 105643. <https://doi.org/10.1016/j.cor.2021.105643>
- [32] Dorigo, M., Maniezzo, V., & Coloni, A. (1996). Ant system: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybernetics*, 26(1), 29–41. <https://doi.org/10.1109/3477.484436>
- [33] Bullnheimer, B., Hartl, R. F., & Strauss, C. (1999). An improved ant system algorithm for the vehicle routing problem. *Annals of Operations Research*, 89, 319–328. <https://doi.org/10.1023/A:1018940026670>
- [34] Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. In *Proceedings of the IEEE International Conference on Neural Networks* (Vol. 4, pp. 1942–1948). IEEE. <https://doi.org/10.1109/ICNN.1995.488968>
- [35] Accorsi, L., & Vigo, D. (2021). A fast and scalable heuristic for the solution of large-scale capacitated vehicle routing problems. *Transportation Science*, 55(4), 832–856. <https://doi.org/10.1287/trsc.2021.1059>
- [36] Bengio, Y., Lodi, A., & Prouvost, A. (2021). Machine learning for combinatorial optimization: A methodological tour d’horizon. *European Journal of Operational Research*, 290(2), 405–421. <https://doi.org/10.1016/j.ejor.2020.07.063>
- [37] Nazari, M., Oroojlooy, A., Snyder, L., & Takáč, M. (2018). Reinforcement learning for solving the vehicle routing problem. *Advances in Neural Information Processing Systems*, 31.

- [38] Kool, W., van Hoof, H., & Welling, M. (2019). Attention, learn to solve routing problems! In *International Conference on Learning Representations*.
- [39] Barr, R. S., Golden, B. L., Kelly, J. P., Resende, M. G. C., & Stewart, W. R. (1995). Designing and reporting on computational experiments with heuristic methods. *Journal of Heuristics*, 1(1), 9–32. <https://doi.org/10.1007/BF02430363>
- [40] Uchoa, E., Pecin, D., Pessoa, A., Poggi, M., Vidal, T., & Subramanian, A. (2017). New benchmark instances for the capacitated vehicle routing problem. *European Journal of Operational Research*, 257(3), 845–858. <https://doi.org/10.1016/j.ejor.2016.08.012>
- [41] López-Ibáñez, M., Dubois-Lacoste, J., Pérez Cáceres, L., Birattari, M., & Stützle, T. (2016). The irace package: Iterated racing for automatic algorithm configuration. *Operations Research Perspectives*, 3, 43–58. <https://doi.org/10.1016/j.orp.2016.09.002>