DOI 10.34658/9788366741102.9

Comparison of Selected Algorithms Solving Vehicle Routing Problem with Simultaneous Delivery and Pickup

Maciej Michalik,
Joanna Ochelska-Mierzejewska[0000−0002−9295−3962]

Lodz University of Technology
Institute of Information Technology
Wólczańska 215, 90-924 Łódź, Poland
joanna.ochelska-mierzejewskai@p.lodz.pl

Abstract. *The Vehicle Routing Problem is a widely known combinatorial optimization problem. A particular variant of this problem is Vehicle Routing Problem with Simultaneous Delivery and Pickup (VRPSDP). In this paper, two metaheuristics are compared in the context of solving the VRPSDP - the Genetic Algorithm (GA) and Ant Colony Optimization (ACO). Both implemented algorithms are hybridized using local search operators. Implemented algorithms are tested using well known Dethloff dataset. The final results show that, in general, ACO gives more accurate results than GA, but it is worse in terms of performance. The main reason for that is the difficulty of incorporating local search operators into the Genetic Algorithm.*
Keywords: *Vehicle Routing Problem, Vehicle Routing Problem with Simultaneous Delivery and Pickup, Genetic Algorithm, Ant Colony Optimization*

1. Introduction

Nowadays, logistics processes play a critical role in sustaining continuous economic growth and supplying goods worldwide. The reason logistics processes are crucial is that they minimize costs related to managing flow and storage of goods. To understand how big the logistics market is, it is worth referring to the forecast provided by PR Newswire, which states that the logistics market size is projected to reach USD 12,236 billion by 2022 [1].

Transportation plays a crucial role in logistics. According to one study, transportation constitutes 30% of all logistics costs [2]. Thus, it is clear that logistics companies are prominently striving to limit expenses related to transportation in order to generate more revenue.

Transportation may consider different modes i.e., railways, waterways, airways, and roadways. This thesis revolves around road transportation only. Thus, further on, only roadways mode is discussed. The costs that mainly affect road transportation are related to:

- fuel prices,
- salaries for drivers,
- vehicle fleet maintenance,
- government regulation,
- geopolitical events,
- regularity of transport.

To minimize the impact of some of the mentioned aspects, companies try to optimize vehicles delivery routes. Such a strategy considers finding delivery routes for a fleet of vehicles so that the total distance covered by all of them is minimal. Reducing the distance covered by vehicles may result in:

- lower fuel consumption,
- lower vehicle exploitation,
- lower required number of vehicles,
- shorter working time of drivers,
- lower costs related to regulations.

Optimizing delivery routes for vehicles might seem an easy task at first. The problem occurs when the number of depots, vehicles, and customers to be served rises, then planning such delivery routes becomes a burdensome task. Such a problem of optimizing vehicles delivery routes is widely known by the name Vehicle Routing Problem (VRP).

This article aims to compare selected algorithms and find answers about their usefulness for the Vehicle Routing Problem with Simultaneous Delivery and Pickup. This comparison will be performed on a known Dethloff dataset to enable comparison of results for data specially prepared for this problem.

Section 2 describes the Vehicle Routing Problem with Simultaneous Delivery and Pickup, discusses the assumptions and presents a mathematical model. Section 3 presents briefly selected algorithms (Genetic Algorithm, Ant Colony Optimization), while section 4 describes Dethloff dataset. Section 5 presents the conducted experiments and also discusses the obtained results.

2. Vehicle Routing Problem with Simultaneous Delivery and Pickup

Numerous VRP variants can be distinguished depending on constraints, objectives, and other factors that affect the problem. To somehow categorize VRP variants and explain the assumptions of VRPSDP, the taxonomy proposed by Braekers, Ramaekers, and Van Nieuwenhuyse [3], is used. The authors suggest using five main characteristics in order to categorize VRPs, i.e., applied methods, scenario characteristics, physical problem characteristics, information characteristics, and data characteristics [3, 4, 5].

The applied method refers to what kind of method is applied to solve a particular problem, e.g., metaheuristic or exact method. The scenario characteristics mainly revolve around constraints that are considered in a given problem, e.g., load splitting constraint or time window structure. Problem physical characteristics consider physical constraints, e.g., number of vehicles or number of depots. Information characteristics indicate what type of information is used in the problem, e.g., static information known upfront or dynamic information. Data characteristics imply what type of data is used, i.e., real-world data, synthetic data, or both.

In the Vehicle Routing Problem with Simultaneous Delivery and Pickup, a homogenous fleet of vehicles operates from a single depot to service all customers who may require both a pickup and delivery demand. The main objective is to minimize the total distance covered by all vehicles while satisfying the following constraints:

- service all customers exactly once,
- all vehicle routes should start and end at the depot,
- all customer deliveries are from the depot,
- all customer pickups must be delivered to the depot,
- at any time, the load should not exceed vehicle capacity,
- the load of the vehicle at the departure from the depot must be equal to the total load for customer deliveries of the corresponding route.

The following figures present an example of VRPSDP and a possible solution to the problem. Node 0 represents depot, and nodes 1 to 4 represent customers, see Figure 1. Each customer has a pickup and delivery demand, and each connection between nodes has a cost which is a distance between nodes.

For the sake of the example, let us assume that there is a single vehicle with a capacity of 120. In the solution presented in Figure 2, the vehicle starts with a load equal to 100 to satisfy all customers delivery demands. The vehicle visits all customers exactly once, and its capacity is never exceeded. The total distance covered might be easily computed by adding costs of a traversed paths, which in the presented example is equal to 184. Of course, this is a trivial example with just a single vehicle. The problem becomes more complicated with the growth of customers to be served and vehicles used.

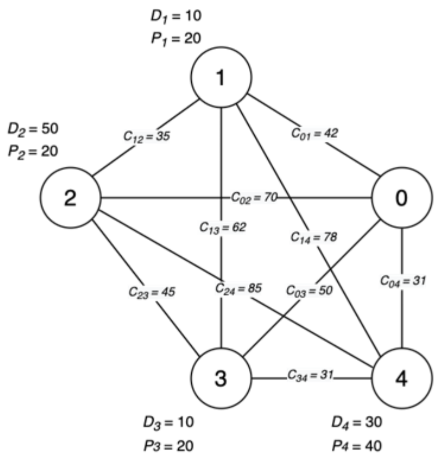

Figure 1. Graphs with solutions for sample SCA3, SCA8, CON3, and CON8 problems generated using Genetic Algorithm without Local Search operators.

2.1. Mathematical Model

The presented mathematical model was proposed by Dethloff, J. in [6].
Notation

Sets

J: Set of all customer nodes
J_0: Set of all nodes including depot

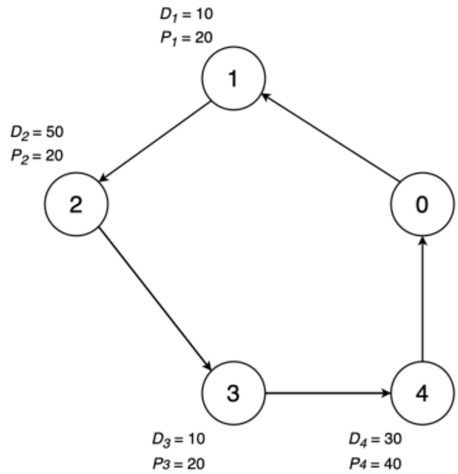

Figure 2. Graphs with solutions for sample SCA3, SCA8, CON3, and CON8 problems generated using Genetic Algorithm without Local Search operators.

V: Set of all vehicles
Parameters
C: Vehicle capacity
D_j: Delivery amount of customer $j \in J$
P_j: Pickup amount of customer $j \in J$
n: Number of nodes, i.e., $n = \|J_0\|$
Decision Variables
l'_v: Load of vehicle v when leaving the depot
l_j: Load of vehicle $j \in J$
n_j: Variable used to prohibit subtours; can be interpreted as position of node $j \in J$ in the route
x_{ijv}: Binary value indicating whether vehicle $v \in V$ travels directly from node $i \in J_0$ to node $j \in J_0$
Model
Minizmize $z = \sum_{i \in J_0} \sum_{j \in J_0} \sum_{v \in V} (C_{ij} x_{ijv})$ — The objective function
$\sum_{iinJ_0} \sum_{v \in V} x_{ijv} = 1; j \in J$ — Each customer is served exactly once
$\sum_{i \in J_0} x_{isv} = \sum_{j \in J_0} x_{sjv}; s \in J, v \in V$ — The same vehicle arrives at and leaves the node
$l'_v = \sum_{i \in J_0} \sum_{j \in J} (D_j x_{ijv}); v \in V$ — The load of the vehicle when leaving the depot must be equal to the total load for customer deliveries of the corresponding route

$l_j \geq l'_v - D_j + P_j - M(1 - x_{0jv}); j \in J, v \in V$ — Vehicle load after visiting first customer
$l'_v \leq C; v \in V$ — Capacity constraints
$l_j \leq C; j \in J$ — Capacity constraints
$n_j \geq 0; j \in J$ — Subtour breaking constraints
$x_{ijv} \in \{0, 1\}; i \in J_0, j \in J_0, v \in V$ — Subtour breaking constraints

3. Algorithms for Solving VRPSDP

The VRP problems are NP-hard problems, meaning that exploring all possible solutions for a particular problem instance is nearly impossible in a reasonable time. In most cases, metaheuristics and heuristics are used to solve such issues [7, 8, 9, 10]. Metaheuristics are strategies that indicate how to perform the search process. Their goal is to explore a large space of possible solutions efficiently to find a near-optimal solution. Metaheuristics are not problem specific. However, different metaheuristics may be better fitted for different types of problems.

In this paper, two main metaheuristics are compared: Genetic Algorithm (GA) and Ant Colony Optimization (ACO). Both these algorithms differ in the way they work. The first one is prevalent and easily adaptable to most optimization problems [11, 12]. The second, on the other hand, is well suited for combinatorial optimization problems [13, 14, 15]. Such a difference would indicate that the ACO is a better solution for VRP problems [16]. On the other hand, GA was proved to be an efficient and accurate solution for many optimization challenges in the past years. Therefore, a comparison of those two algorithms should be intriguing.

Additionally, each of the main algorithms is hybridized using local search algorithms. Local search algorithms are used to improve already existing solutions in an iterative manner. Local search algorithms might be applied to both ACO and GA as both of these algorithms produce solutions that might be enhanced. The following three local search algorithms are discussed and elaborated on in the next chapters: 2-opt operator, Swap operator, and CROSS-exchange operator [4].

4. Benchmark Dataset

In order to examine and compare implemented algorithms, the Dethloff dataset is used [6]. This is one of the most commonly used datasets for benchmarking VRPSDP algorithms. This dataset provides synthetic data, meaning depot and customers are placed on the cartesian plane. Also, the delivery and pick-up values, as well as vehicle capacity, are integer values with no unit.

The dataset proposed by Dethloff consists of four scenarios, each including ten randomly generated test cases, resulting in 40 test cases overall. The four scenarios are:

- SCA3 — customers are scattered uniformly on the plane, the minimum number of vehicles to be used is 3 and capacity of vehicles is high.
- SCA8 — customers are scattered uniformly on the plane, the minimum number of vehicles to be used is 8 and capacity of vehicles is low.
- CON3 — half of customers concentrated/clustered other half uniformly scattered, the minimum number of vehicles to be used is 3 and capacity of vehicles is high.
- CON8 — half of customers concentrated/clustered other half uniformly scattered, the minimum number of vehicles to be used is 8 and capacity of vehicles is low.

Each test case consists of 50 customers and a single depot. The vehicle capacity is calculated based on the minimum number of vehicles to be used - the less the minimum number of vehicles, the higher the capacity and vice-versa.

5. Experiments

Ten different variants of algorithms are examined. Each of the variants is used to solve all 40 test cases from the Dethloff dataset. In total, 400 experiments are conducted. The average objective function results and average execution times are collected for every scenario, i.e., SCA3, SCA8, CON3, and CON8, for every algorithm variant.

For the Genetic Algorithm, the following variants are examined:

- Genetic Algorithm without Local Search operators;
- Genetic Algorithm with 2-opt Local Search operator;
- Genetic Algorithm with Swap Local Search operator;
- Genetic Algorithm with CROSS-Exchange Local Search operator;
- Genetic Algorithm with all Local Search operators.

For the Ant Colony Optimization algorithm, the following variants are examined:

- ACO algorithm without Local Search operators;
- ACO algorithm with 2-opt Local Search operator;
- ACO algorithm with Swap Local Search operator;
- ACO algorithm with CROSS-Exchange Local Search operator;
- ACO algorithm with all Local Search operators.

In order to save time and not conduct too many experiments required for finding optimal parameters, the parameters found in the literature were used in both the Ant Colony Optimization algorithm and the Genetic Algorithm.

Parameters Selection for Genetic Algorithm

In the Genetic Algorithm implementation proposed in [17], the authors suggest that best results are achieved for population size 150 and number of generations 300. For the crossover, the probability of 0.8 is recommended, and for mutation, the probability of 0.03. Also, in the Genetic Algorithm introduced in this paper, the 5 elitist individuals are copied from the current generation to the next.

Parameters Selection for Ant Colony Optimization

In the Ant Colony Optimization algorithm implementation proposed in [18], the α and β coefficients are both set to 5, the number of elitist ants is 10, evaporation factor is 0.05, and the maximum number of iterations is 100.

Parameters for Local Search Operators

There are no parameters to be set for a 2-opt search. For both the Swap operator and CROSS-exchange operator, the K parameter denoting the number of nearest neighbours is set to 10.

5.1. Genetic Algorithm without Local Search operators

Table 1 presents the results of the Genetic Algorithm without local search operators applied. For each scenario SCA3, SCA8, CON3, CON8, the average results were calculated using ten problem instances. Also, Figure 3 shows graphs with solutions for sample SCA3, SCA8, CON3, and CON8 problem instances. The Genetic Algorithm without local search operators gives the worst objective function results from all tested algorithm variants. Even though computations for this algorithm are relatively fast, they are not the fastest, as Ant Colony Optimization without local search operators is faster.

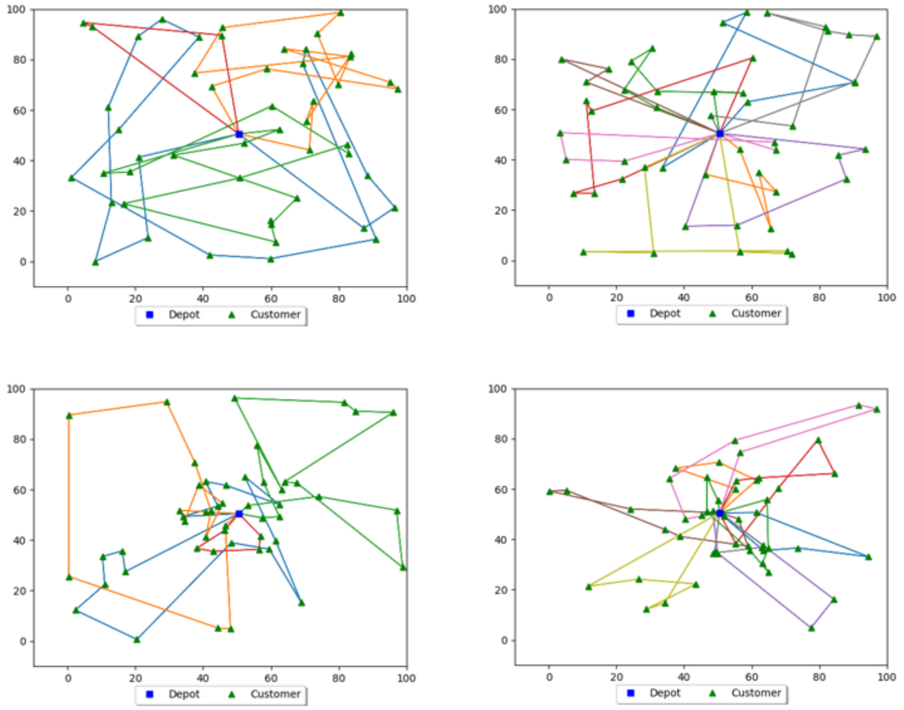

Figure 3. Graphs with solutions for sample SCA3, SCA8, CON3, and CON8 problems generated using Genetic Algorithm without Local Search operators.

Table 1. Results for Genetic Algorithm without Local Search operators.

Problem Instance	Execution Time (s)	Objective Function Result
SCA3	10.00	1272.70
SCA8	8.64	1572.43
CON3	9.93	966.65
CON8	8.40	1132.65
Average	9.24	1236.11

5.2. Genetic Algorithm with 2-opt Local Search operator

Table 2 presents the results of the Genetic Algorithm with 2-opt local search operator applied. For each scenario SCA3, SCA8, CON3, CON8, the average results were calculated using ten problem instances. Also, Figure 4 shows graphs with solutions for sample SCA3, SCA8, CON3, and CON8 problem instances. Using the Genetic Algorithm with 2-opt operator results in significant enhancement of the objective function results, comparing to the previously mentioned GA variant. At the same time, the execution time has not increased much.

Table 2. Results for Genetic Algorithm with 2-opt Local Search operator.

Problem Instance	Execution Time (s)	Objective Function Result
SCA3	11.49	1001.39
SCA8	8.26	1469.90
CON3	11.82	803.93
CON8	8.28	1074.93
Average	9.96	1087.54

5.3. Genetic Algorithm with Swap Local Search operator

Table 3 presents the results of the Genetic Algorithm with Swap local search operator applied. For each scenario SCA3, SCA8, CON3, CON8, the average results were calculated using ten problem instances. Also, Figure 5 shows graphs with solutions for sample SCA3, SCA8, CON3, and CON8 problem instances. For the Genetic Algorithm variant with Swap local search operator, the results are enhanced. However, not as much as for the GA with the 2-opt operator variant. Also, applying the Swap operator causes execution time to extend significantly.

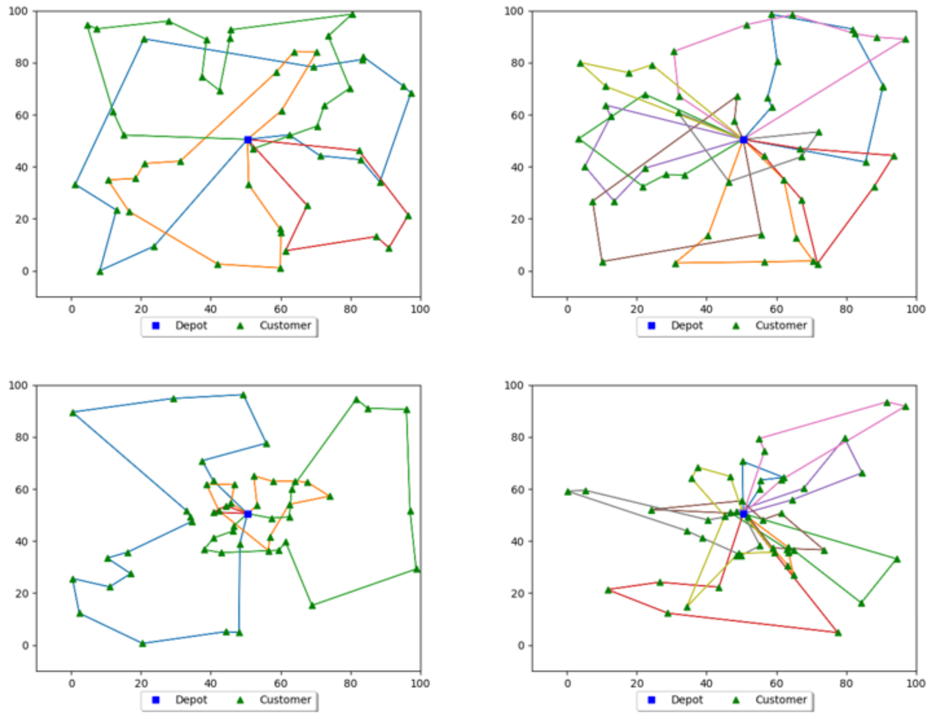

Figure 4. Graphs with solutions for sample SCA3, SCA8, CON3, and CON8 problems generated using Genetic Algorithm with 2-opt Local Search operator.

Table 3. Results for Genetic Algorithm with Swap Local Search operator.

Problem Instance	Execution Time (s)	Objective Function Result
SCA3	34.05	1155.33
SCA8	27.95	1462.01
CON3	35.19	902.03
CON8	28.80	1034.67
Average	31.50	1138.51

5.4. Genetic Algorithm with CROSS-Exchange Local Search operator

Table 4 presents the results of the Genetic Algorithm with CROSS-Exchange local search operator applied. For each scenario SCA3, SCA8, CON3, CON8, the average results were calculated using ten problem instances. Also, Figure 6 shows graphs with solutions for sample SCA3, SCA8,

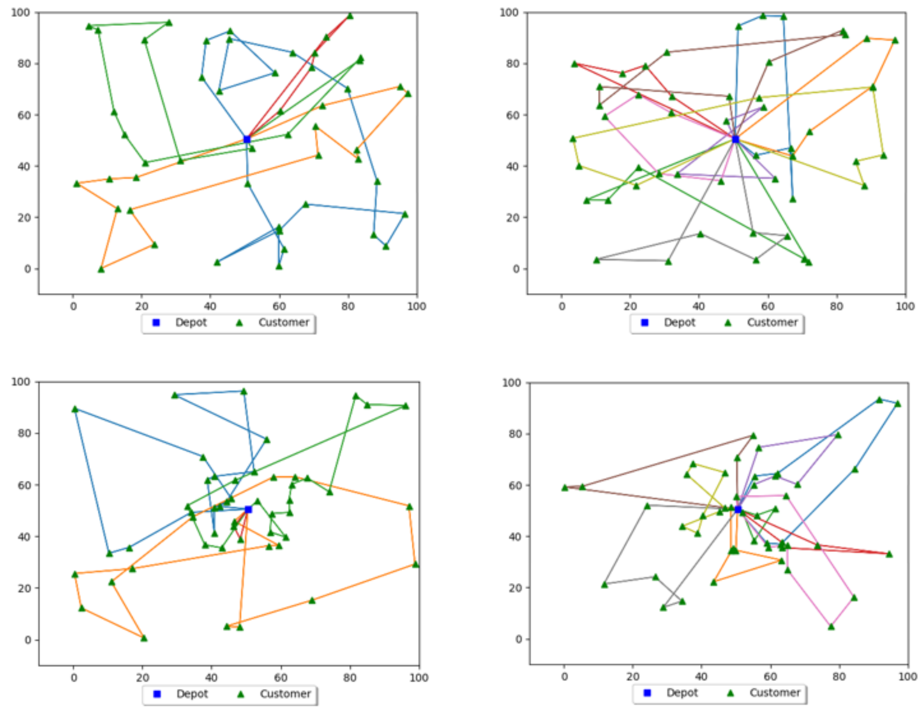

Figure 5. Graphs with solutions for sample SCA3, SCA8, CON3, and CON8 problems generated using Genetic Algorithm with Swap Local Search operator.

CON3, and CON8 problem instances. For the Genetic Algorithm variant with CROSS-Exchange local search operator, the results are enhanced, but not as much as for GA variants with 2-opt and Swap operators. Also, applying the CROSS-Exchange operator causes execution time to extend even more comparing to GA with Swap local search operator.

Table 4. Results for Genetic Algorithm with CROSS-Exchange Local Search.

Problem Instance	**Execution Time (s)**	**Objective Function Result**
SCA3	197. 63	1163.17
SCA8	23.16	1490.20
CON3	204.23	899.41
CON8	24.54	1074.76
Average	112.39	1156.88

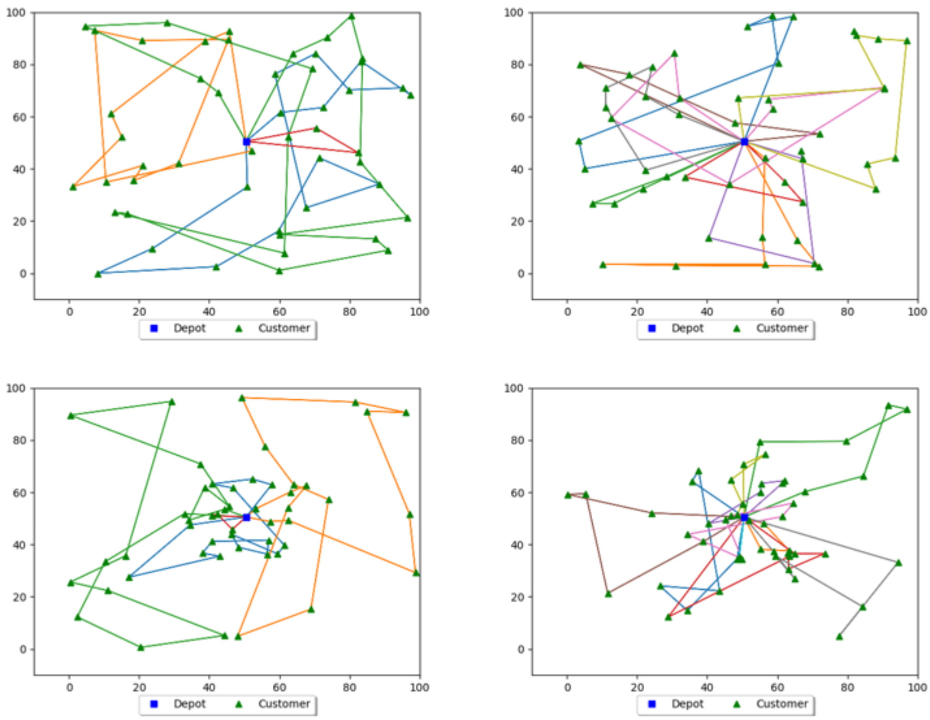

Figure 6. Graphs with solutions for sample SCA3, SCA8, CON3, and CON8 problems generated using Genetic Algorithm with CROSS-Exchange Local Search operator.

5.5. Genetic Algorithm with all Local Search operators

Table 5 presents the results of the Genetic Algorithm with all local search operators applied. For each scenario SCA3, SCA8, CON3, CON8, the average results were calculated using ten problem instances. Also, Figure 7 shows graphs with solutions for sample SCA3, SCA8, CON3, and CON8 problem instances. The Genetic Algorithm with all local search operators applied gives the most accurate solutions from all GA variants. Also, the execution time was the longest among all GA variants.

Table 5. Results for Genetic Algorithm with all Local Search operators.

Problem Instance	Execution Time (s)	Objective Function Result
SCA3	215.36	876.66
SCA8	42.097	1348.96
CON3	213.98	707.96
CON8	45.98	967.48
Average	129.35	975.27

5.6. Ant Colony Optimization without Local Search operators

Table 6 presents the results of the Ant Colony Optimization without local search operators applied. For each scenario SCA3, SCA8, CON3, CON8, the average results were calculated using ten problem instances. Also, Figure 8 shows graphs with solutions for sample SCA3, SCA8, CON3, and CON8 problem instances. The Ant Colony Optimization without local search operators gives the worst objective function results among all ACO variants. At the same time, it is the fastest among all tested variants.

Table 6. Results for Ant Colony Optimization without Local Search operators.

Problem Instance	Execution Time (s)	Objective Function Result
SCA3	2.90	1110.13
SCA8	7.09	1491.63
CON3	2.83	908.41
CON8	7.40	1128.73
Average	5.05	1159.72

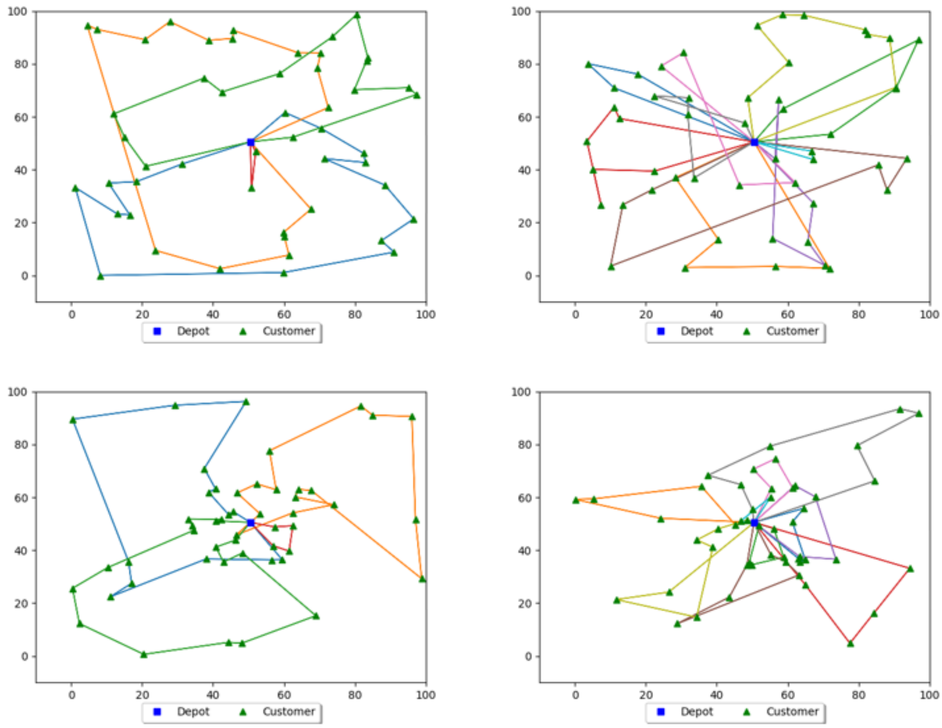

Figure 7. Graphs with solutions for sample SCA3, SCA8, CON3, and CON8 problems generated using Genetic Algorithm with all Local Search operators.

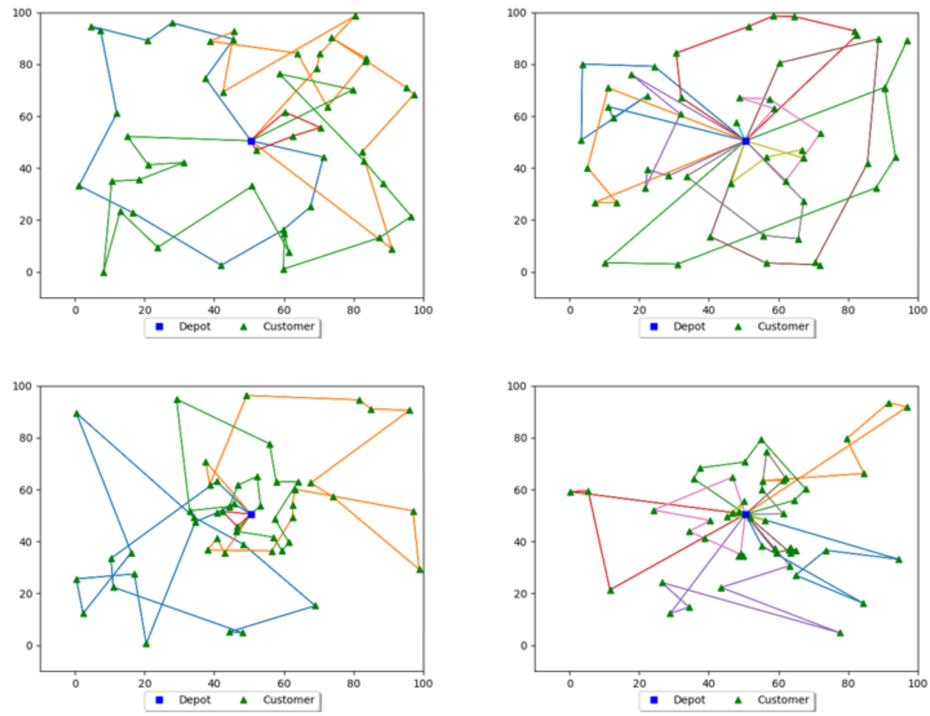

Figure 8. Graphs with solutions for sample SCA3, SCA8, CON3, and CON8 problems generated using Ant Colony Optimization without Local Search operators.

5.7. Ant Colony Optimization with 2-opt Local Search operator

Table 7 presents the results of the Ant Colony Optimization with 2-opt local search operator applied. For each scenario SCA3, SCA8, CON3, CON8, the average results were calculated using ten problem instances. Also, Figure 9 shows graphs with solutions for sample SCA3, SCA8, CON3, and CON8 problem instances. As with the Genetic Algorithm, the ACO variant with 2-opt operator applied results in evident enhancements of the accuracy, keeping the execution time at a relatively low level.

Table 7. Results for Ant Colony Optimization with 2-opt Local Search operator.

Problem Instance	Execution Time (s)	Objective Function Result
SCA3	13.25	876.95
SCA8	9.59	1298.90
CON3	13.81	743.72
CON8	9.86	1014.88
Average	11.63	983.62

5.8. Ant Colony Optimization with Swap Local Search operator

Table 8 presents the results of the Ant Colony Optimization with Swap local search operator applied. For each scenario SCA3, SCA8, CON3, CON8, the average results were calculated using ten problem instances. Also, Figure 10 shows graphs with solutions for sample SCA3, SCA8, CON3, and CON8 problem instances. The ACO variant with the Swap operator gives more accurate results than the ACO with no local search operator applied, but not as accurate results as the ACO with the 2-opt local search operator. Also, using the Swap operator extends the execution time significantly.

5.9. Ant Colony Optimization with CROSS-Exchange Local Search operator

Table 9 presents the results of the Ant Colony Optimization with CROSS-Exchange local search operator applied. For each scenario SCA3, SCA8, CON3, CON8, the average results were calculated using ten problem instances. Also, Figure 11 shows graphs with solutions for sample SCA3, SCA8, CON3, and CON8 problem instances. For the ACO variant with

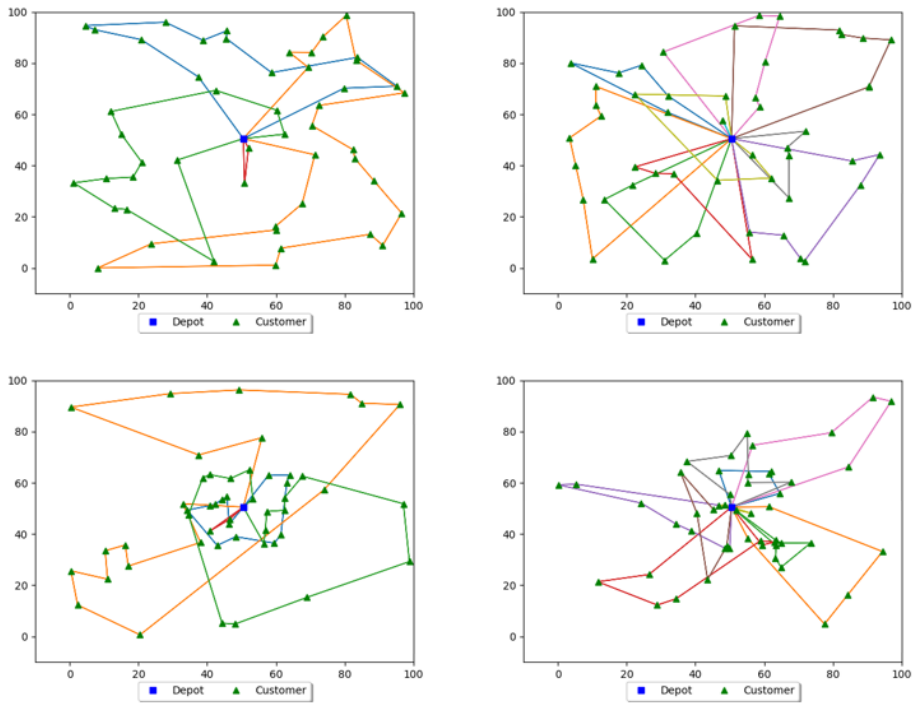

Figure 9. Graphs with solutions for sample SCA3, SCA8, CON3, and CON8 problems generated using Ant Colony Optimization with 2-opt Local Search operator.

Table 8. Results for Ant Colony Optimization with Swap Local Search operator.

Problem Instance	**Execution Time (s)**	**Objective Function Result**
SCA3	78.45	969.50
SCA8	189.85	1313.01
CON3	77.07	831.43
CON8	183.78	977.20
Average	132.29	1022.79

CROSS-Exchange local search operator, the results are enhanced, but not as much as for ACO variants with 2-opt and Swap operators. Also, applying the CROSS-Exchange operator causes execution time to extend even more comparing to ACO with Swap local search operator.

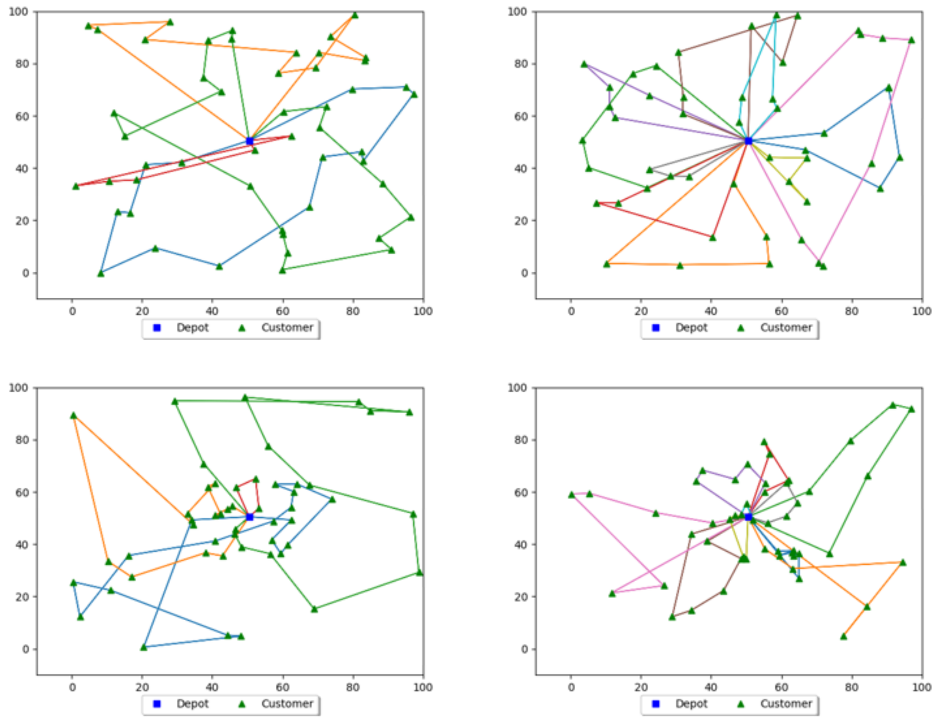

Figure 10. Graphs with solutions for sample SCA3, SCA8, CON3, and CON8 problems generated using Ant Colony Optimization with Swap Local Search operator.

Table 9. Results for Ant Colony Optimization with CROSS-Exchange Local Search operator.

Problem Instance	Execution Time (s)	Objective Function Result
SCA3	554.07	1045.88
SCA8	125.25	1410.83
CON3	500.46	862.99
CON8	137.21	1073.73
Average	329.26	1098.36

5.10. Ant Colony Optimization with all Local Search operators

Table 10 presents the results of the Ant Colony Optimization with all local search operators applied. For each scenario SCA3, SCA8, CON3, CON8, the average results were calculated using ten problem instances.

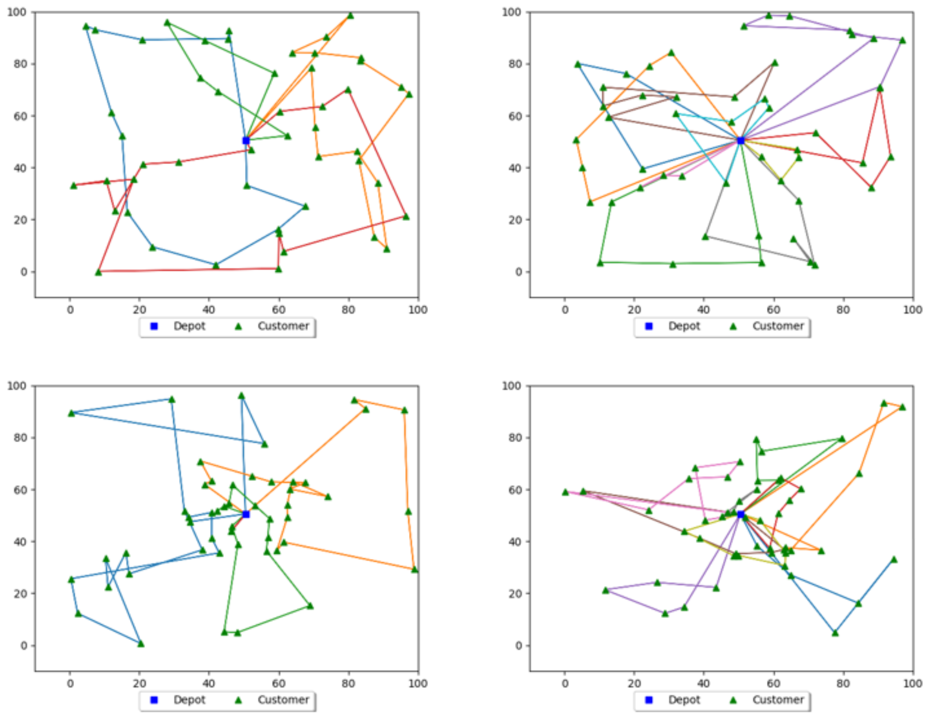

Figure 11. Graphs with solutions for sample SCA3, SCA8, CON3, and CON8 problems generated using Ant Colony Optimization with CROSS-Exchange Local Search operator.

Also, Figure 12 shows graphs with solutions for sample SCA3, SCA8, CON3, and CON8 problem instances. The Ant Colony Optimization with all local search operators gives the best objective function results among all tested variants. At the same time, it is the slowest tested algorithm.

Table 10. Results for Ant Colony Optimization with all Local Search operators.

Problem Instance	**Execution Time (s)**	**Objective Function Result**
SCA3	622.71	787.52
SCA8	297.36	1230.81
CON3	601.20	667.24
CON8	337.67	921.74
Average	464.73	901.83

6. Disscusion and Conclusions

Genetic Algorithm vs Ant Colony Optimization The Genetic Algorithm performed worse than the Ant Colony Optimization in terms of accuracy of results but generally better in terms of execution time. Also, local search operators had a more significant impact on the Ant Colony Optimization than the Genetic Algorithm if it comes to accuracy and execution time.

Local Search Operators Among three tested local search operators, the 2-opt operator had the most significant impact on the accuracy of both the ACO and the GA. At the same time, it affected the execution time the least.

The Swap local search operator performed a little better than the CROSS-Exchange operator in terms of enhancing the accuracy of the algorithm but still worse than the 2-opt operator. Applying the Swap operator to both the ACO and the GA resulted in slower execution time. Comparing to variants without local search operators, for GA, it took three times longer to complete the algorithm, and for the ACO, 26 times longer.

The CROSS-Exchange operator performed the worst out of three tested local search operators. Not only did it enhance the objective function results the least, but it also extended the execution time the most.

In general, the Genetic Algorithm performed worse than the Ant Colony Optimization in terms of solving VRPSDP. The reason for that mainly lies in the difficulty of hybridizing the Genetic Algorithm with local search operators. Applying the local search operators to offspring generated by genetic operators is hugely time-consuming. The reason for that is the way

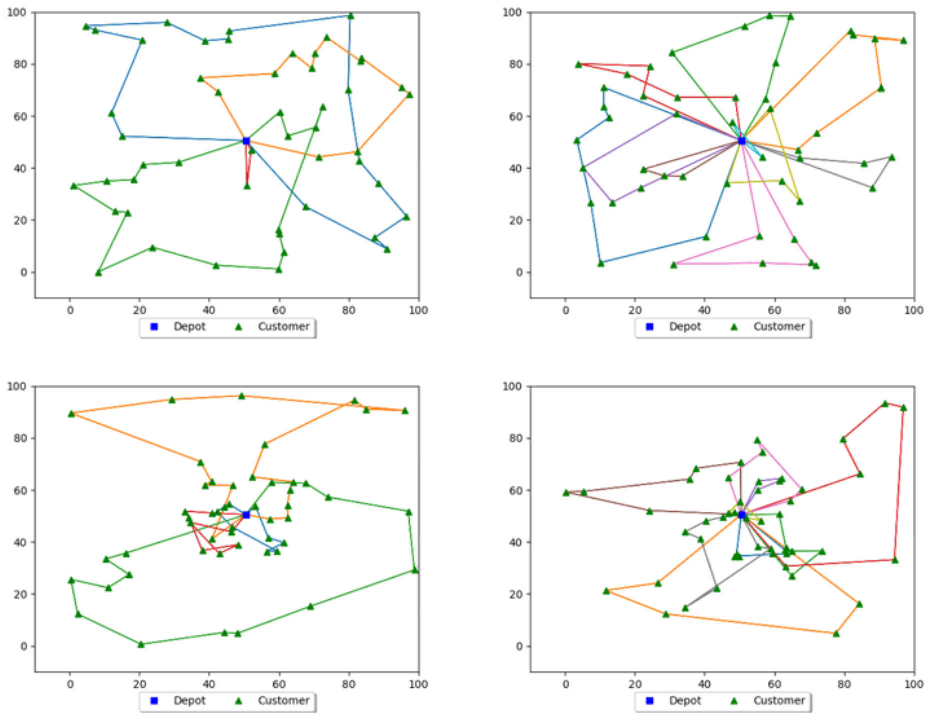

Figure 12. Graphs with solutions for sample SCA3, SCA8, CON3, and CON8 problems generated using Ant Colony Optimization with all Local Search operators.

PMX crossover and Swap mutation genetic operators work. The offspring may consist of entirely disordered solutions. The more the routes for vehicles are disordered, the more neighborhood solutions are considered for Swap and CROSS-Exchange local search operators. For that reason, the local search operators were only applied once the Genetic Algorithm produced the final population. In order to incorporate the local search operators into the process of generating the new population, one should minimize mutation probability and find a more complex crossover operator that would produce more ordered solutions. To sum up, the Genetic Algorithm is not the best fit for solving VRPSDP.

On the other hand, the Ant Colony Optimization performed very well with local search operators, which should not be surprising as ACO algorithms usually use local search strategies for enhancing solutions. The local search operators work so well for ACO because the solutions produced by ants are generally quite well-ordered. Thus, when Swap and CROSS-Exchange local search operators were applied, they did not create so many neighborhood solutions, which had to be considered. It is also worth mentioning that the best results found to date for the Dethloff dataset were achieved using ACO with well-adjusted search operators.

As it comes to local search operators, the 2-opt operator is the most impactful. Mainly because it enhances solutions accuracy the most, and it is quite fast. It is faster than Swap and CROSS-exchange operators because it generates fewer neighborhood solutions that need to be considered. The Swap and CROSS-Exchange operators also enhance the final solutions, but they require much more time to finish the search process. When implementing these operators, one needs to consider limiting the neighborhood solutions by manipulating the parameter K.

References

[1] Reports, V. Logistics market size is projected to reach usd 12,256 billion by 2022 - valuates reports, 2020. URL `https://www.prnewswire.com/news-releases/logistics-market-size-is-projected-to-reach-usd-12-256-billion-by-2022---valuates-reports-301065076.html`.

[2] Tseng, Y., Yue, W., and Taylor, M. A. P. The role of transportation in logistics chain. In *Proceedings of the Eastern Asia Society for Transportation Studies*, volume 5. 2005.

[3] K., K. B., Ramaekers, K., and van Nieuwenhuyse, I. The vehicle routing problem: State of the art classification and review. *Computers & Industrial Engineering*, 99:300–313, 2001. doi:10.1016/j.cie.2015.12.007. URL `https://www.sciencedirect.com/science/article/pii/S0360835215004775`.

[4] Arnold, F. and Sörensen, K. Knowledge-guided local search for the vehicle routing problem. *Computers & Operations Research*, 105:32–46, 2019. doi:10.1016/j.cor.2019.01.002. URL `https://www.sciencedirect.com/science/article/pii/S0305054819300024`.

[5] Bertsimas, D. and Ryzin, G. V. A stochastic and dynamic vehicle routing problem in the euclidean plane. *Operations Research*, 39(4):601–615, 1991. doi:10.1287/opre.39.4.601.

[6] Dethloff, J. Vehicle routing and reverse logistics: The vehicle routing problem with simultaneous delivery and pick-up. *OR-Spektrum*, 23(1):79–96, 2001. doi:10.1007/PL00013346. URL `https://link.springer.com/article/10.1007/PL00013346`.

[7] Nagy, G. and Salhi, S. Heuristic algorithms for single and multiple depot vehicle routing problems with pickups and deliveries. *European Journal of Operational Research*, 162(1):126–141, 2005. doi:10.1016/j.ejor.2002.11.003. URL `https://www.sciencedirect.com/science/article/pii/S0377221703008361`.

[8] Blum, C. and Roli, A. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Computing Surveys*, 35(3):268–308, 2003. doi:10.1145/937503.937505.

[9] Christofides, N., Mingozzi, A., and Toth, P. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Computing Surveys*, 35(3):268–308, 2003. doi:10.1145/937503.937505.

[10] Ochelska-Mierzejewska, J. *Tabu Search Algorithm for Vehicle Routing Problem with Time Windows*, pages 117–136. Springer International Publishing, 2020. doi:10.1007/978-3-030-34706-2.

[11] Sivanandam, S. N. and Deepa, S. N. *Introduction to Genetic Algorithms.* Springer-Verlag, Berlin Heidelberg, 2008.

[12] Gen, M., Cheng, R., and Lin, L. *Network Models and Optimization: Multiobjective Genetic Algorithm Approach.* Springer-Verlag, London, 2008.

[13] Dorigo, M. and Stützle, T. The ant colony optimization metaheuristic: Algorithms, applications, and advances. In *Handbook of Metaheuristics*, page 250–285. MA: Springer US (International Series in Operations Research & Management Science), 2003. doi:10.1007/0-306-48056-5_9.

[14] Ant colony optimization algorithms, 2020. URL `https://en.wikipedia.org/w/index.php?title=Ant\_colony\_optimization\_al\-gorithms\&oldid=995540400`.

[15] Ochelska-Mierzejewska, J. Ant colony optimization algorithm for split delivery vehicle routing problem. In *Advanced Information Networking and Applications*, pages 758–767. 2020.

[16] Kalayci, C. and Kaya, C. An ant colony system empowered variable neighborhood search algorithm for the vehicle routing problem with simultaneous pickup and delivery. *Expert Systems with Applications*, 66:163–175, 2016. doi:10.1016/j.eswa.2016.09.017. URL https://www.sciencedirect.com/science/article/pii/S0957417416304961.

[17] Tasan, A. and Gen, M. A genetic algorithm based approach to vehicle routing problem with simultaneous pick-up and deliveries. *Computers & Industrial Engineering*, 62(3):755–761, 2012. doi:10.1016/j.cie.2011.11.025. URL https://www.sciencedirect.com/science/article/pii/S0360835211003482.

[18] Gajpal, Y. and Abad, P. An ant colony system (acs) for vehicle routing problem with simultaneous delivery and pickup. *Computers & Operations Research*, 36(12):3215–3223, 2009. doi:10.1016/j.cor.2009.02.017. URL https://www.sciencedirect.com/science/article/pii/S0305054809000537.