

See discussions, stats, and author profiles for this publication at: <https://www.researchgate.net/publication/372763935>

A Comparative Analysis of Search Algorithms for Solving the Vehicle Routing Problem

Chapter · July 2023

DOI: 10.5772/intechopen.112067

CITATIONS

0

READS

64

1 author:



[Samuel Oladimeji Sowole](#)

African Institute for Mathematical Sciences Senegal

7 PUBLICATIONS 43 CITATIONS

SEE PROFILE

We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

6,500

Open access books available

175,000

International authors and editors

190M

Downloads

Our authors are among the

154

Countries delivered to

TOP 1%

most cited scientists

12.2%

Contributors from top 500 universities



WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index
in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com



A Comparative Analysis of Search Algorithms for Solving the Vehicle Routing Problem

Oladimeji Samuel Sowole

Abstract

The Vehicle Routing Problem (VRP) is an extensively studied optimization challenge in operations research, applicable to logistics, transportation, and supply chain management. Its goal is to find optimal routes for vehicles, minimizing distance and maximizing customer satisfaction. Genetic algorithms, simulated annealing, and ant colony optimization are search algorithms commonly used to solve the VRP. This chapter provides a comparative analysis of these algorithms, highlighting their strengths and weaknesses. It introduces the VRP and its variants, along with associated challenges and constraints, and offers an overview of different search algorithms used for solving the problem, explaining their principles, advantages, and limitations. Real-world case studies showcase successful applications of these algorithms in package delivery, waste collection, and emergency response. Additionally, the chapter explores key factors influencing algorithm performance, including problem size, complexity, and parameters. It concludes by providing recommendations for selecting appropriate algorithms for different VRP instances. By providing a comprehensive understanding of search algorithms for the VRP, this chapter enables readers to make informed decisions when addressing similar optimization problems in practical scenarios.

Keywords: vehicle routing problem (VRP), search algorithms, optimization, metaheuristics, ant colony optimization (ACO), particle swarm optimization (PSO), genetic algorithms (GA), tabu search, simulated annealing, heuristics, routing strategies

1. Introduction

The Vehicle Routing Problem (VRP) is a well-known optimization problem in operations research, with significant applications in logistics, transportation, and supply chain management. It involves determining the optimal routes for a fleet of vehicles to serve a set of customers while minimizing costs and maximizing customer satisfaction. Efficiently solving the VRP has substantial implications for reducing transportation costs, improving resource utilization, and enhancing overall operational efficiency [1–3].

1.1 Problem statement and objectives

The VRP can be formally defined as follows:

Given:

1. A set of customers, denoted by $C = \{c_1, c_2, \dots, c_n\}$
2. A depot, denoted by D
3. A fleet of vehicles, denoted by $V = \{v_1, v_2, \dots, v_m\}$
4. Each customer c_i has a demand d_i , representing the quantity of goods to be delivered
5. Each vehicle v_i has a capacity q , representing the maximum amount it can carry
6. The distance between any two locations i and j is represented by $d(i, j)$.

The objective of the VRP is to find:

- For each vehicle v_i , a route denoted by R_i , starting and ending at the depot D
- The sequence of customers to be visited along each route

The objectives of the VRP are to:

- Minimize the total distance traveled by all vehicles
- Minimize the number of vehicles used
- Balance the workload among the vehicles
- Satisfy the demand of each customer.

1.2 Definition and variants of VRP

The VRP encompasses various problem variants, each introducing additional constraints or objectives. Some common variants of the VRP include:

1.2.1 Capacitated VRP (CVRP)

In Capacitated VRP (CVRP), each vehicle has a limited capacity, and the total demand served by each vehicle must not exceed its capacity [3]. **Figure 1** gives an illustration of CVRP. Mathematically, CVRP can be formulated as follows:

Minimize:

$$\sum d(i, j) * x(i, j) \quad (1)$$

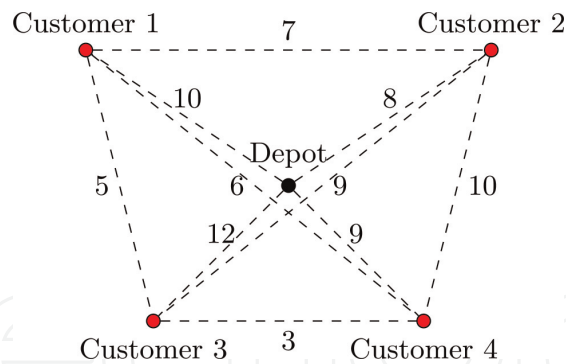


Figure 1.
 Capacitated vehicle routing problem (CVRP) illustration.

Subject to:

1. $\sum x(i,j) = 1$, for all $i \in C, j \in C, i \neq j$
2. $\sum x(j,i) = 1$, for all $i \in C, j \in C, i \neq j$
3. $\sum x(i,j) \leq m * y(i)$, for all $i \in C, j \in C, i \neq j$
4. $\sum x(j,i) \leq m * y(i)$, for all $i \in C, j \in C, i \neq j$
5. $\sum y(i) = m$
6. $\sum d(i) * y(i) \leq q$, for all $i \in C$
7. $x(i,j) \in \{0, 1\}$, for all $i \in C, j \in C, i \neq j$
8. $y(i) \in \{0, 1\}$, for all $i \in C$.

Where:

- $x(i,j)$ is a binary decision variable denoting whether customer i is assigned to customer j in the solution
- $y(i)$ is a binary decision variable denoting whether customer i is visited in the solution
- $d(i,j)$ represents the distance between customers i and j
- q is the capacity of each vehicle
- m is the total number of vehicles.

1.2.2 Vehicle routing problems with time windows (VRPTW)

For Vehicle Routing Problems with Time Windows (VRPTW), customers have specified time windows during which they can be served, and the vehicles must respect these time constraints [4]. **Figure 2** gives a schematic illustration of VRPTW.

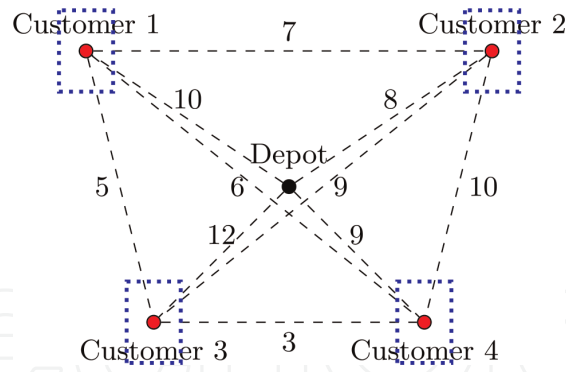


Figure 2.
Vehicle routing problem with time windows (VRPTW) illustration.

Mathematically, Vehicle Routing Problem with Time Windows (VRPTW) can be formulated as follows:

Minimize:

$$\sum d(i,j) * x(i,j) \quad (2)$$

Subject to:

1. $\sum x(i,j) = 1$, for all $i \in C, j \in C, i \neq j$
2. $\sum x(j,i) = 1$, for all $i \in C, j \in C, i \neq j$
3. $\sum x(i,j) \leq m * y(i)$, for all $i \in C, j \in C, i \neq j$
4. $\sum x(j,i) \leq m * y(i)$, for all $i \in C, j \in C, i \neq j$
5. $\sum y(i) = m$
6. $\sum d(i) * y(i) \leq q$, for all $i \in C$
7. $a(i) \leq t(i) \leq b(i) - d(i,j) - M * (1 - x(i,j))$, for all $i \in C, j \in C, i \neq j$
8. $x(i,j) \in \{0, 1\}$, for all $i \in C, j \in C, i \neq j$
9. $y(i) \in \{0, 1\}$, for all $i \in C$.

Where:

- $x(i,j)$ is a binary decision variable denoting whether customer i is assigned to customer j in the solution
- $y(i)$ is a binary decision variable denoting whether customer i is visited in the solution
- $d(i,j)$ represents the distance between customers i and j
- q is the capacity of each vehicle

- m is the total number of vehicles
- $a(i)$ and $b(i)$ represent the start and end of the time window for customer i , respectively
- $t(i)$ represents the arrival time at customer i
- M is a large positive constant.

1.3 Main challenges and constraints

The VRP presents several challenges and constraints that need to be considered in the solution process. Some of the main challenges and constraints include:

1. **Capacity Constraint:** Each vehicle has a limited capacity, and the total demand served by each vehicle must not exceed its capacity. Mathematically, this constraint can be represented as: $\sum d(i,j) * y(i,j) \leq q$, for all $i \in C, j \in C, i \neq j$.
2. **Time Window Constraint:** In VRPTW, customers have specified time windows during which they can be served. The vehicles must respect these time constraints and ensure that the arrival time at each customer falls within its time window. This constraint can be expressed as: $a(i) \leq t(i) \leq b(i) - d(i,j) - M * (1 - x(i,j))$, for all $i \in C, j \in C, i \neq j$. Where $a(i)$ and $b(i)$ represent the start and end of the time window for customer i , respectively, $t(i)$ is the arrival time at customer i , $d(i,j)$ is the travel time between customers i and j , and M is a large positive constant.
3. **Routing Constraints:** Each vehicle must start and end its route at the depot, and it should visit each customer exactly once. Additionally, the routes should not contain subcycles or disconnected components.
4. **Objective Function:** The objective is to minimize the total distance traveled by all vehicles while satisfying the demand of each customer and respecting the capacity and time window constraints.

Solving the VRP considering these challenges and constraints requires the application of efficient search algorithms. In the following section, we will explore and analyze different search algorithms, namely genetic algorithms, simulated annealing, and ant colony optimization, which have been widely used to tackle the VRP.

2. Search algorithms for VRP

2.1 Genetic algorithms

Genetic algorithms (GAs) are population-based search algorithms inspired by the process of natural evolution. They utilize genetic operators, such as selection,

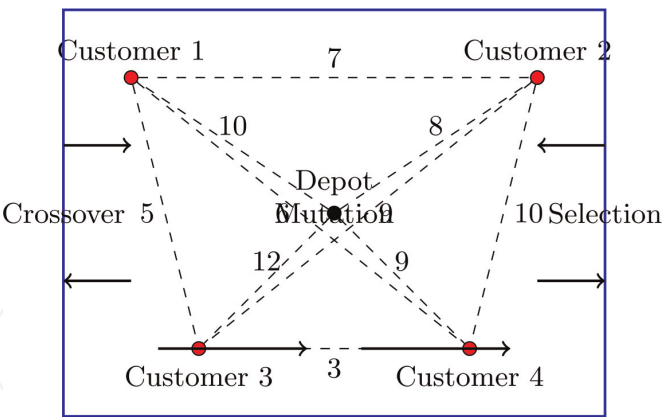


Figure 3.
Genetic algorithm (GA) for vehicle routing problem (VRP) illustration.

crossover, and mutation, to iteratively evolve a population of candidate solutions toward better solutions. GAs have been widely applied to solve various optimization problems, including the VRP [5, 6]. We give an illustration of GA in **Figure 3**.

2.1.1 Genetic representation and operators

In the context of the VRP, a common representation of a solution is a chromosome, which encodes the sequence of customers to be visited by a vehicle. The genetic operators used in GAs for the VRP are as follows:

1. Initialization: Generate an initial population of random feasible solutions.
2. Fitness evaluation: Evaluate the fitness of each solution based on the objective function (e.g., total distance traveled).
3. Selection: Select a subset of solutions from the population based on their fitness values, giving preference to better solutions.
4. Crossover: Combine pairs of selected solutions to create offspring solutions. This involves exchanging segments of routes between parents.
5. Mutation: Randomly modify some elements (e.g., customer sequence) in the offspring solutions to introduce diversity.
6. Replacement: Replace some solutions in the population with the newly created offspring solutions.
7. Termination: Repeat the above steps until a termination condition is met (e.g., a maximum number of generations or a convergence criterion).

The genetic operators allow exploration of the solution space by preserving good characteristics from the parent solutions and introducing variation through crossover and mutation.

2.2 Fitness evaluation

The fitness function in GAs for the VRP measures the quality of a solution based on the objectives of minimizing the total distance traveled, balancing the workload among vehicles, and satisfying the capacity constraints. The fitness evaluation typically involves calculating the total distance traveled by each vehicle, considering the sequence of customers visited.

2.2.1 Mathematical formulation

Given a solution represented by a chromosome, the fitness function can be defined as:

$$\text{Fitness} = (\alpha * \text{TotalDistance}) + (\beta * \text{WorkloadImbalance}) + (\gamma * \text{CapacityViolation}) \quad (3)$$

where:

α , β , and γ are weight coefficients that balance the importance of each objective. The total distance can be calculated by summing the distances between consecutive customers visited by each vehicle. Workload imbalance can be quantified by measures such as the standard deviation of the vehicle workloads or the maximum workload difference. Capacity violation accounts for the amount by which the demand exceeds the vehicle capacity.

2.2.2 Advantages and limitations

The advantages of using genetic algorithms for the VRP include the:

- Ability to explore a large solution space efficiently.
- Ability to handle different problem variants and constraints.
- Potential to find high-quality solutions.

However, GAs also have some limitations:

- Computational complexity increases with problem size, limiting their applicability to large-scale instances.
- Difficulty in striking a balance between exploration and exploitation.
- Selection of appropriate parameter values can be challenging.

2.3 Simulated annealing

Simulated annealing (SA) is a stochastic search algorithm that emulates the annealing process in metallurgy. It uses a probabilistic acceptance criterion to accept worse solutions early in the search process and gradually focuses on improving the objective function [7, 8]. **Figure 4** gives a schematic illustration of SA.

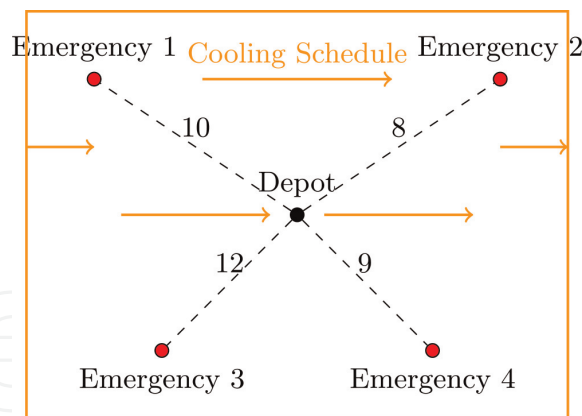


Figure 4.
Simulated annealing (SA) for vehicle routing problem (VRP) illustration.

2.3.1 Annealing schedule and neighborhood structure

In SA for the VRP, an initial solution is generated, and a temperature parameter is set. The algorithm iteratively explores the solution space by perturbing the current solution and accepting new solutions based on a probability function that depends on the objective function difference and the current temperature [7, 8]. The neighborhood structure defines the set of feasible moves from the current solution. In the VRP, common neighborhood structures include swapping customers between routes or inserting a customer from one route into another route while maintaining the capacity and time window constraints.

2.3.2 Acceptance criteria

The acceptance criterion in SA determines whether to accept a new solution based on its objective function value and the current temperature. Initially, the algorithm allows a higher probability of accepting worse solutions, which facilitates exploration. As the temperature decreases, the algorithm gradually becomes more selective and focuses on improving the objective function.

2.3.3 Mathematical formulation

Given the current solution with objective function value $F(\text{current})$ and a new solution with objective function value $F(\text{new})$, the acceptance probability can be defined as:

$$P = \frac{\exp((F(\text{current}) - F(\text{new})))}{T}, \tag{4}$$

where T is the current temperature.

The acceptance probability allows the algorithm to accept worse solutions with a higher probability at the beginning, gradually reducing the probability as the temperature decreases.

2.3.4 Advantages and limitations

Advantages of simulated annealing for the VRP include:

- Ability to escape local optima by allowing moves that worsen the objective function.

- Flexibility in exploring the solution space through the acceptance probability.
- Effective for problems with a complex search landscape.

However, SA also has some limitations:

- Dependence on parameter tunings, such as the initial temperature and the cooling schedule.
- Convergence to suboptimal solutions if the cooling schedule is not appropriately chosen.
- Computationally intensive for large problem instances.

2.4 Ant colony optimization

Ant Colony Optimization (ACO) is a metaheuristic algorithm inspired by the foraging behavior of ants. It utilizes pheromone-based communication and stigmergy to iteratively construct high-quality solutions to the VRP. **Figure 5** is a schematic illustration of ACO.

2.4.1 Pheromone-based communication and stigmergy

In ACO for the VRP, artificial ants construct solutions by iteratively moving from one customer to another, depositing pheromones along their paths [9]. The concentration of pheromone on an edge represents the desirability of that edge. Ants probabilistically choose the next customer to visit based on the pheromone levels and heuristic information (e.g., distance) of the neighboring customers. Pheromone evaporation and reinforcement mechanisms ensure the dynamic update of pheromone trails. Evaporation reduces the pheromone levels over time to avoid convergence to suboptimal solutions, while reinforcement strengthens the pheromone trail of good-quality solutions found by the ants [9, 10].

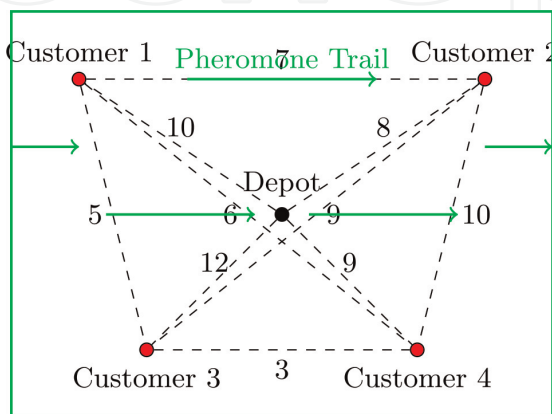


Figure 5.
Ant colony optimization (ACO) for vehicle routing problem (VRP) illustration.

2.4.2 Construction of routes and pheromone update

In ACO, each ant constructs a complete set of routes by iteratively selecting customers to visit based on the pheromone and heuristic information. The pheromone update is performed by evaporating the existing pheromone and depositing the pheromone on the edges traversed by the ants based on the quality of the solution found.

2.4.3 Mathematical formulation

The pheromone update rule can be expressed as:

$$\tau(i,j) = (1 - \rho) * \tau(i,j) + \sum (\Delta\tau_k(i,j)) \quad (5)$$

Where:

$\tau(i,j)$ represents the pheromone level on the edge (i,j) , ρ is the evaporation rate, and $\Delta\tau_k(i,j)$ represents the pheromone deposit made by ant k on the edge (i,j) .

The pheromone deposit $\Delta\tau_k(i,j)$ can be defined as:

$$\Delta\tau_k(i,j) = \frac{Q}{Lk}, \quad (6)$$

where Q is a constant representing the pheromone quantity deposited by each ant and Lk is the objective function value (e.g., total distance) of the solution constructed by ant k .

2.4.4 Advantages and limitations

Advantages of Ant Colony Optimization for the VRP include:

- Robustness and adaptability to dynamic problem instances and changes in the solution space.
- Effective exploration of the solution space through the probabilistic decision-making process.
- Ability to capture and exploit good solutions found by the ants through pheromone reinforcement.
- Scalability to handle large-scale instances of the VRP.

However, ACO also has some limitations:

- Convergence to suboptimal solutions if the algorithm gets trapped in local optima.
- Sensitivity to parameter settings, such as the evaporation rate and pheromone deposit constant.
- Difficulty in handling complex constraints, especially with multiple objectives.

2.5 Other search algorithms for VRP

In addition to Genetic Algorithms, Simulated Annealing, and Ant Colony Optimization, several other search algorithms have been applied to solve the VRP. Here are some notable examples:

2.6 Tabu search

Tabu Search is a metaheuristic algorithm that employs a memory-based search strategy to explore the solution space [11]. It maintains a short-term memory called the tabu list, which prohibits recently visited solutions from being revisited. Tabu Search iteratively explores the neighborhood of the current solution, allowing moves that improve the objective function or violate certain tabu conditions [11].

2.6.1 Particle swarm optimization

Particle Swarm Optimization (PSO) is a population-based optimization algorithm inspired by the social behavior of bird flocking or fish schooling. In PSO, candidate solutions (particles) move through the solution space based on their own best-known position and the best-known position of the swarm. PSO aims to converge to optimal solutions by iteratively updating the velocities and positions of the particles.

2.6.2 Iterated local search

Iterated Local Search (ILS) is a metaheuristic algorithm that combines local search with perturbation and diversification [12]. It starts with an initial solution and applies a local search algorithm to improve it. After reaching a local optimum, ILS introduces perturbations to escape the local optima and continues the search. The process of local search and perturbation is repeated iteratively to improve the solution quality [12].

2.6.3 Hybrid approaches

Hybrid approaches combine multiple search algorithms or integrate heuristic rules with optimization algorithms to enhance the performance of VRP solvers [13]. For example, a hybrid algorithm may incorporate a construction heuristic to generate initial solutions, followed by a local search algorithm or a metaheuristic to refine the solutions.

These are just a few examples of the numerous search algorithms that have been explored for solving the VRP. Each algorithm has its strengths and limitations, making them suitable for different problem instances and objectives. The choice of the search algorithm depends on the specific requirements and constraints of the VRP instance at hand.

In the following section, we will present case studies and examples of how these search algorithms have been applied to solve the VRP in real-world scenarios, highlighting their performance and applicability.

3. Case studies and examples

In this section, we will explore various case studies and examples of using search algorithms to solve the Vehicle Routing Problem (VRP) in different real-world scenarios. We will discuss the problem formulation and algorithm implementation, and evaluate the performance of the algorithms in each case.

3.1 Vehicle routing for package delivery

Package delivery is a common application of the VRP, where a fleet of vehicles is responsible for delivering packages to customers efficiently. Let us consider a case where there are multiple vehicles and a set of customers with specific demands and time windows.

3.1.1 Problem formulation

Consider a scenario where a package delivery company needs to optimize the routes for a fleet of vehicles to deliver packages to a set of customers. Each customer has a specific location, package demand, and delivery time window. The objective is to minimize the total distance traveled by vehicles while ensuring that all packages are delivered within their respective time windows and that the vehicle capacity is not exceeded. Let:

- Set of customers $C = \{c_1, c_2, \dots, c_n\}$ with demands d_1, d_2, \dots, d_n
- Set of vehicles $V = \{v_1, v_2, \dots, v_m\}$ with capacities q_1, q_2, \dots, q_m
- Distance matrix D representing the distances between customers and depots
- Location of each customer $c \in C$ represented by coordinates (x_c, y_c)
- Package demand for each customer $c \in C$ denoted by d_c
- Time window for each customer $c \in C$ specified as $[earliest_{time_c}, latest_{time_c}]$
- Vehicle capacity for each vehicle $v \in V$ denoted by q_v

Objective:

Minimize the total distance traveled by all vehicles while satisfying the package delivery time windows and vehicle capacity constraints.

The mathematical formulation for the Vehicle Routing Problem for Package Delivery can be represented as given below.

Now, let us denote the decision variables:

- $x_{cv} = 1$ if vehicle v visits customer c , 0 otherwise.
- d_{cd} = Distance between customer c and customer d .
- t_{cv} = Time at which vehicle v arrives at customer c .

The mathematical formulation for the Vehicle Routing for Package Delivery problem can then be represented as:

Minimize:

$$\sum_{v \in V} \sum_{c \in C} \sum_{d \in C} (d_{cd} * x_{cvc} * x_{dv}) \quad (7)$$

Subject to:

1. $\sum_{v \in V} \sum_{c \in C} (x_{cvc} * d_c) \leq q_v$, for all $v \in V$ (Vehicle capacity constraint)
2. $\sum_{v \in V} (x_{cvc}) = 1$, for all $c \in C$ (Each customer is visited exactly once)
3. $\sum_{c \in C} (x_{cvc}) = 1$, for all $v \in V$ (Each vehicle visits exactly one customer)
4. $t_{cv} \geq \frac{t_c + d_c}{V_{min}}$, for all $v \in V, c \in C$ (Time window constraint—Arrival time at each customer must be after its earliest time window)
5. $t_{cv} \leq \frac{t_c + d_c}{V_{max}}$, for all $v \in V, c \in C$ (Time window constraint—Arrival time at each customer must be before its latest time window)
6. $\sum_{c \in C} (t_{cv} * x_{cvc}) \leq T_v$, for all $v \in V$ (Total time traveled by each vehicle must be within its maximum time limit)
7. $x_{cvc} \in \{0, 1\}$, for all $c \in C, v \in V$ (Binary variable indicating whether a vehicle visits a customer)

The objective function aims to minimize the total distance traveled by all vehicles. It sums up the distances between customers c and d , multiplied by the binary variables x_{cvc} and x_{dv} , which indicate whether a vehicle visits those customers.

The first constraint ensures that the demand of each customer does not exceed the capacity of the vehicle assigned to it. It sums up the demands d_c at each customer c visited by vehicle v , multiplied by the binary variable x_{cvc} , and enforces that it must be less than or equal to the capacity q_v of vehicle v .

The second constraint ensures that each customer is visited exactly once. It states that the sum of binary variables x_{cvc} over all vehicles v must be equal to 1 for each customer c .

The third constraint ensures that each vehicle visits exactly one customer. It states that the sum of binary variables x_{cvc} over all customers c must be equal to 1 for each vehicle v .

The fourth and fifth constraints enforce the time window constraints. They state that the arrival time t_{cv} at each customer c by vehicle v must be after its earliest time window (t_c) and before its latest time window (t_c). The travel time d_c divided by the maximum and minimum vehicle speeds (V_{max} and V_{min}) is added to the customer's time window.

The sixth constraint ensures that the total time traveled by each vehicle, considering the arrival times at each customer is within its maximum time limit T_v .

Finally, the binary variables x_{cvc} are constrained to be either 0 or 1, indicating whether a vehicle visits a customer.

By solving this mathematical formulation using appropriate search algorithms, an optimized set of routes for the package delivery vehicles can be obtained, minimizing the total distance traveled while satisfying the capacity and time window constraints. Here, we show how to implement this with Genetic Algorithm.

3.1.2 Algorithm implementation: Genetic algorithm

Here's a step-by-step implementation of the Genetic Algorithm for solving the VRP for package delivery:

Step 1: Initialization

- Generate an initial population of solutions, where each solution represents a set of routes for the vehicles.
- The routes should ensure that all customers are visited exactly once and satisfy the capacity and time window constraints.

Step 2: Fitness evaluation

- Evaluate the fitness of each solution in the population.
- The fitness can be determined by the total distance traveled, considering any penalties for violating time window constraints or exceeding vehicle capacities.

Step 3: Selection

- Select parent solutions from the population for the crossover operation.
- The selection can be based on fitness values, employing techniques such as tournament selection or roulette wheel selection.

Step 4: Crossover

- Apply crossover operators, such as ordered crossover or partially mapped crossover, to create offspring solutions.
- The crossover should ensure that the offspring inherit good characteristics from the parent solutions.

Step 5: Mutation

- Apply mutation operators, such as swap mutation or insertion mutation, to introduce diversity in the population.
- The mutation helps explore new areas of the search space and avoid premature convergence.

Step 6: Fitness evaluation

- Evaluate the fitness of the offspring solutions.

Step 7: Replacement

- Select solutions from the population, including both parents and offspring, to form the new population.
- The replacement can follow strategies like elitism or generational replacement.

Step 8: Termination condition

- Check if the termination condition is met, such as reaching a maximum number of generations or a specific fitness threshold.
- If the condition is met, stop the algorithm; otherwise, go back to Step 2.

Step 9: Output the best solution

- Select the solution with the highest fitness from the final population as the best solution.
- This solution represents the optimized routes for the vehicles to deliver packages.

We give the algorithm implementation of the steps above:

3.2 Vehicle routing for waste collection

Waste collection is another important application of the VRP, where vehicles are assigned to collect waste from various locations efficiently. Let us consider a case where there are multiple waste collection points.

3.2.1 Problem formulation

Consider a scenario where waste collection vehicles need to be routed efficiently to collect waste from various locations. Each location has a specific waste quantity, and there are capacity constraints on the vehicles. The objective is to minimize the total distance traveled by vehicles while ensuring that the waste collection demand is met and the vehicle capacity is not exceeded.

Let:

- Set of waste collection points, $P = \{p_1, p_2, \dots, p_n\}$
- Set of vehicles $V = \{v_1, v_2, \dots, v_m\}$
- Location of each waste collection point $p \in P$ represented by coordinates (x_p, y_p)
- Waste quantity at each waste collection point $p \in P$ denoted by q_p
- Vehicle capacity for each vehicle $v \in V$ denoted by Q_v

Objective:

Minimize the total distance traveled by all vehicles while satisfying the waste collection demand and vehicle capacity constraints.

Let us denote the decision variables:

$x_{pvp} = 1$ if vehicle v visits waste collection point p , 0 otherwise.

d_{pq} = Euclidean distance between waste collection points p and q .

The mathematical formulation for the Vehicle Routing for Waste Collection problem can then be represented as:

Minimize:

$$\sum_{v \in V} \sum_{p \in P} \sum_{q \in P} (d_{pq} * x_{pvp} * x_{qv}) \quad (8)$$

Subject to:

1. $\sum_{v \in V} \sum_{p \in P} \sum_{q \in P} (q_p * x_{pvp}) \leq Q_v$, for all $v \in V$ (Vehicle capacity constraint)
2. $\sum_{v \in V} x_{pvp} = 1$, for all $p \in P$ (Each waste collection point is visited exactly once)
3. $\sum_{p \in P} x_{pvp} = 1$, for all $v \in V$ (Each vehicle visits exactly one waste collection point)
4. $x_{pvp} \in \{0, 1\}$, for all $p \in P, v \in V$ (Binary variable indicating whether a vehicle visits a waste collection point)

The objective function aims to minimize the total distance traveled by all vehicles. It sums up the distances between waste collection points p and q , multiplied by the binary variables x_{pvp} and x_{qv} , which indicate whether a vehicle visits those points.

The first constraint ensures that the waste collection demand at each waste collection point does not exceed the capacity of the vehicle assigned to it. It sums up the waste quantities q_p at each waste collection point p visited by vehicle v , multiplied by the binary variable x_{pvp} , and enforces that it must be less than or equal to the capacity Q_v of vehicle v .

The second constraint ensures that each waste collection point is visited exactly once. It states that the sum of binary variables x_{pvp} over all vehicles v must be equal to 1 for each waste collection point p .

The third constraint ensures that each vehicle visits exactly one waste collection point. It states that the sum of binary variables x_{pvp} over all waste collection points p must be equal to 1 for each vehicle v .

Finally, the binary variables x_{pvp} are constrained to be either 0 or 1, indicating whether a vehicle visits a waste collection point.

Solving this mathematical formulation using appropriate search algorithms can provide an optimized set of routes for the waste collection vehicles, minimizing the total distance traveled while meeting waste collection demand and vehicle capacity constraints. In this chapter, we give an example of using the Simulated Annealing algorithm to solve this.

3.2.2 Algorithm implementation: Ant colony optimization

Here's a step-by-step implementation of the Ant Colony Optimization (ACO) algorithm for solving the VRP for waste collection:

Step 1: Initialization

- Initialize the pheromone trails between waste collection points and depots.
- Set the parameters, such as the number of ants, maximum iterations, and pheromone evaporation rate.

Step 2: Ant solution construction:

Repeat the following steps for each ant:

- Choose a depot as the starting point.
- While there are unvisited waste collection points:
 - Select the next waste collection point to visit based on the pheromone levels and heuristic information.
 - Update the capacity of the ant's vehicle.
 - Update the pheromone trail based on the visited waste collection point and the distance traveled.

Step 3: Update pheromone trails

- Update the pheromone trails based on the solutions constructed by the ants.
- The pheromone update can consider the quality of the solutions, such as the total distance traveled or the waste collection demand fulfilled.

Step 4: Termination condition

- Check if the termination condition is met, such as reaching a maximum number of iterations or a specific improvement threshold.
- If the condition is met, stop the algorithm; otherwise, go back to Step 2.

Step 5: Output the best solution

- Select the best solution found during the iterations.
- This solution represents the optimized routes for the waste collection vehicles.

3.3 Vehicle routing for emergency response

3.3.1 Problem formulation

Consider a scenario where emergency response vehicles need to be dispatched efficiently to emergency incidents with varying priorities and response time requirements. The objective is to minimize the total distance traveled by vehicles while considering the incident priorities and response time requirements.

Let:

- Set of emergency incidents $E = \{e_1, e_2, \dots, e_n\}$
- Set of vehicles $V = \{v_1, v_2, \dots, v_m\}$
- Location of each emergency incident $e \in E$ represented by coordinates (x_e, y_e)
- Priority of each emergency incident $e \in E$ denoted by p_e
- Response time requirement for each emergency incident $e \in E$ denoted by r_e
- capacity for each vehicle $v \in V$ denoted by Q_v .

Objective:

Minimize the total distance traveled by all vehicles while considering the incident priorities and response time requirements.

We shall now write the formulation mathematically.

Let us denote the decision variables:

$x_{eve} = 1$ if vehicle v is dispatched to emergency incident e , 0 otherwise.

d_{ef} = Euclidean distance between emergency incidents e and f .

The mathematical formulation for the Vehicle Routing for Waste Collection problem can be represented as follows:

Minimize:

$$\sum_{v \in V} \sum_{e \in E} \sum_{f \in E} (d_{ef} * x_{eve} * x_{fv}) \quad (9)$$

Subject to:

1. $\sum_{v \in V} \sum_{e \in E} (p_e * x_{eve}) \leq Q_v$, for all $v \in V$ (Vehicle capacity constraint)
2. $\sum_{v \in V} (x_{eve}) = 1$, for all $e \in E$ (Each emergency incident is assigned exactly one vehicle)
3. $\sum_{e \in E} (x_{eve}) \leq 1$, for all $v \in V$ (Each vehicle is assigned to at most one emergency incident)
4. $\sum_{v \in V} \sum_{e \in E} (r_e * x_{eve}) \leq T_{max}$, for all $v \in V$ (Total response time of each vehicle must be within the maximum time limit)
5. $x_{eve} \in 0, 1$, for all $e \in E, v \in V$ (Binary variable indicating whether a vehicle is dispatched to an emergency incident)

The objective function aims to minimize the total distance traveled by all vehicles. It sums up the distances between emergency incidents e and f , multiplied by the binary variables x_{eve} and x_{fv} , which indicate whether a vehicle is dispatched to those incidents.

The first constraint ensures that the sum of incident priorities p_e at each emergency incident e dispatched to by vehicle v , multiplied by the binary variable x_{eve} , does not exceed the capacity Q_v of vehicle v .

The second constraint ensures that each emergency incident is assigned exactly one vehicle. It states that the sum of binary variables x_{eve} over all vehicles v must be equal to 1 for each emergency incident e .

The third constraint ensures that each vehicle is assigned to at most one emergency incident. It states that the sum of binary variables x_{eve} over all emergency incidents e must be less than or equal to 1 for each vehicle v .

The fourth constraint ensures that the total response time of each vehicle, considering the response time requirements r_e at each emergency incident, is within the maximum time limit T_{max} .

Finally, the binary variables x_{eve} are constrained to be either 0 or 1, indicating whether a vehicle is dispatched to an emergency incident or not.

By solving this mathematical formulation using appropriate search algorithms, an optimized set of dispatch routes for the emergency response vehicles can be obtained, minimizing the total distance traveled while considering incident priorities and response time requirements. Here, we give an algorithm implementation using Simulated Annealing.

3.3.2 Algorithm implementation: Simulated annealing

Here's a step-by-step implementation of the Simulated Annealing algorithm for solving the VRP for emergency response:

Step 1: Initialization

- Generate an initial solution, where each vehicle is assigned a set of emergency incidents to respond to.
- The solution should satisfy the capacity constraints and the response time requirements.

Step 2: Define objective function

- Define an objective function that combines the total distance traveled by vehicles with penalties for violating the response time requirements.

Step 3: Set initial temperature and cooling schedule

- Set an initial temperature for the simulated annealing algorithm.
- Define a cooling schedule to reduce the temperature over iterations.

Step 4: Iterate until termination condition.

While the termination condition is not met:

- Generate a neighboring solution by applying a neighborhood operator, such as swapping two incidents between vehicles or reassigning an incident to a different vehicle.
- Calculate the objective function difference between the current solution and the neighboring solution.

- Determine whether to accept the neighboring solution based on the objective function difference and the current temperature.
- Update the current solution if the neighboring solution is accepted.
- Update the temperature according to the cooling schedule.

Step 5: Output the best solution

- Select the best solution found during the iterations.
- This solution represents the optimized assignment of emergency incidents to vehicles.

3.4 Comparison of algorithm performance in different scenarios

To compare the performance of the search algorithms (Genetic Algorithm, Simulated Annealing, and Ant Colony Optimization) in different VRP scenarios, we can consider the following evaluation metrics:

I.Objective function value:

1. Calculate the objective function value (e.g., total distance traveled) for each algorithm in various problem instances.
2. Compare the objective function values to determine which algorithm yields better solutions.

II.Computational time:

1. Measure the computational time required for each algorithm to solve different problem instances.
2. Analyze the efficiency and scalability of the algorithms based on their computational time.

III.Solution quality:

1. Assess the quality of the solutions generated by each algorithm by comparing them to known optimal solutions or lower bounds.
2. Use metrics such as the percentage deviation from the optimal solution or the solution quality relative to a lower bound.

IV.Robustness:

1. Evaluate the robustness of each algorithm by considering how well they perform in the face of variations in problem parameters, such as the number of customers, vehicle capacities, or time window constraints.

2. Analyze the algorithms' ability to handle different problem instances with varying complexities.

V. Sensitivity analysis:

1. Conduct sensitivity analysis to examine the algorithms' sensitivity to changes in algorithm parameters, such as mutation rate or cooling schedule.
2. Identify the parameter settings that result in optimal performance for different scenarios.

VI. Visualization:

1. Visualize the routes and assignments generated by each algorithm to gain insights into their behavior and effectiveness.
2. Use graphical representations, such as route maps or heatmaps of pheromone trails, to compare the algorithm's solutions visually.

By comparing the algorithm performance in different scenarios using these evaluation metrics, we can assess their strengths, weaknesses, and suitability for specific VRP instances. The analysis will help in selecting the most appropriate algorithm based on the problem characteristics and requirements.

4. Factors influencing algorithm performance

In this section, we will discuss the key factors that influence the performance of search algorithms for solving the Vehicle Routing Problem (VRP). Understanding these factors is crucial for selecting the most appropriate algorithm and optimizing its performance.

4.1 Problem size and complexity

The size and complexity of the VRP instance play a significant role in determining the performance of search algorithms. The problem size refers to the number of customers or waste collection points, while complexity refers to the intricacy of the constraints and objectives involved.

As the problem size increases, the search space grows exponentially, making it more challenging to find an optimal solution within a reasonable amount of time. The complexity of the problem, such as the presence of time windows, vehicle capacities, or precedence constraints, further adds to the computational burden.

Search algorithms may struggle to find optimal solutions for large and complex VRP instances due to the increased search space and computational requirements. However, some algorithms, like genetic algorithms or ant colony optimization, are designed to handle larger problem sizes and complex constraints more effectively.

4.2 Algorithm parameters

The performance of search algorithms is influenced by various parameters that control their behavior and exploration-exploitation trade-off. It is essential to understand the impact of these parameters and tune them appropriately for different VRP instances.

Some common algorithm parameters include population size, mutation rate, crossover rate, pheromone evaporation rate, and cooling schedule. The values of these parameters can significantly impact the convergence speed, solution quality, and exploration capabilities of the algorithm.

For example, a higher mutation rate in genetic algorithms can promote exploration but may slow down convergence. On the other hand, a lower mutation rate can lead to premature convergence and suboptimal solutions. Similarly, the cooling schedule in simulated annealing determines the balance between exploration and exploitation.

Optimal parameter settings for search algorithms depend on the problem characteristics and the desired trade-offs between solution quality and computation time. Experimentation and parameter tuning techniques, such as grid search or evolutionary algorithms, can help identify suitable parameter values for specific VRP instances.

4.3 Problem-specific constraints

The VRP exhibits various constraints that must be considered during algorithm design and implementation. These constraints may include vehicle capacities, time windows, precedence relationships, or specific customer requirements.

The presence of constraints affects the search space and solution feasibility. For example, the vehicle capacity constraint ensures that the total demand of the assigned customers does not exceed the vehicle's capacity. The time window constraint requires deliveries to be made within specific time intervals.

Search algorithms need to incorporate these constraints in their formulations, fitness evaluation functions, and solution construction mechanisms. Violation of constraints often incurs penalties or leads to infeasible solutions. Therefore, algorithms must balance the exploration of the search space while ensuring constraint satisfaction.

Different algorithms may handle constraints differently. For example, genetic algorithms can use repair mechanisms to fix infeasible solutions, while ant colony optimization can enforce constraints during route construction by considering pheromone trails and heuristics.

Understanding the problem-specific constraints and selecting an algorithm that can handle them effectively are crucial for achieving feasible and optimized solutions in VRP.

4.4 Computational resources

The computational resources available, such as processing power, memory, and time, have a significant impact on algorithm performance. Search algorithms with high computational requirements may not be feasible to run on limited resources or within tight time constraints.

Large-scale and complex VRP instances may require significant computational resources to explore the vast search space and find optimal solutions. In such cases,

parallel computing techniques or distributed algorithms can be employed to leverage multiple processors or computing nodes.

Consideration should also be given to the trade-off between solution quality and computational time. Some instances may require near-optimal solutions, while others may prioritize finding good-quality solutions within a reasonable time frame. To address the influence of computational resources on algorithm performance, several strategies can be employed:

I. Algorithm optimization

1. Enhance the efficiency of the algorithm by implementing algorithmic optimizations, such as pruning techniques, heuristic initialization, or local search operators.
2. Streamline the algorithm's operations to reduce computational overhead and improve convergence speed.

II. Problem decomposition

1. Decompose the problem into smaller subproblems that can be solved independently or in parallel.
2. Divide the VRP instance into clusters or zones and assign different computing resources to solve each subproblem concurrently.
3. Combine the solutions from the subproblems to obtain a solution for the entire problem.

III. Approximation algorithms

1. Utilize approximation algorithms that provide near-optimal solutions with lower computational requirements.
2. Approximation algorithms trade off optimality for computational efficiency and can be well suited for large-scale VRP instances.

IV. Metaheuristic hybridization

1. Combine different search algorithms or metaheuristics to leverage their strengths and compensate for their weaknesses.
2. Hybrid algorithms can take advantage of the exploration capabilities of one algorithm and the exploitation capabilities of another to improve overall performance.

V. Resource scaling

1. Scale the problem instance or adjust the level of detail to match the available computational resources.

2. For example, if the problem size is too large to solve optimally, consider aggregating or sampling the data to reduce the computational requirements while maintaining the problem's essential characteristics.

VI. Hardware and parallel computing

1. Utilize high-performance computing resources, such as multicore processors, distributed computing clusters, or cloud computing platforms, to accelerate algorithm execution.
2. Implement parallelization techniques, such as parallel algorithms, parallel fitness evaluation, or parallel solution construction, to exploit the available computing resources efficiently.

It is crucial to consider the computational resources when selecting an algorithm for VRP and to assess the algorithm's performance within the constraints of available resources. By optimizing algorithms, utilizing decomposition techniques, employing approximation methods, or leveraging parallel computing, it becomes possible to overcome limitations imposed by computational resources and find efficient solutions for VRP instances.

5. Recommendations for algorithm selection

Selecting the appropriate algorithm for solving the Vehicle Routing Problem (VRP) depends on the characteristics of the problem instance and the desired objectives. In this section, we provide recommendations for algorithm selection based on specific VRP scenarios.

5.1 Small-scale VRP instances

For small-scale VRP instances with a relatively small number of customers and simple constraints, exact algorithms, such as integer linear programming (ILP) formulations or branch-and-bound methods, can be effective. These algorithms aim to find the globally optimal solution by exploring the entire search space.

5.1.1 ILP formulation

1. Formulate the VRP as an ILP problem using binary decision variables to represent the assignment of customers to vehicles.
2. Define objective function and constraints, including vehicle capacity, time windows, and precedence relationships.
3. Use ILP solvers, such as CPLEX or Gurobi, to solve the formulation and obtain an optimal solution.

5.1.2 Branch-and-bound method

1. Employ a tree-based search approach to explore the search space systematically.

2. Divide the problem into subproblems and use lower bounds to prune infeasible or non-promising branches.
3. Continually refine the lower bounds and update the best solution found during the search process.

5.2 Large-scale VRP instances

For large-scale VRP instances with a high number of customers and complex constraints, metaheuristic algorithms are more suitable due to their ability to handle larger search spaces efficiently.

Genetic algorithms, ant colony optimization, and particle swarm optimization are popular metaheuristic algorithms for large-scale VRP instances. These algorithms can explore the search space effectively and provide near-optimal solutions within a reasonable computational time.

In addition, decomposition techniques, such as clustering or partitioning the problem into smaller subproblems, can be applied to further enhance the scalability of the algorithms.

5.3 VRP instances with time windows

When dealing with VRP instances that have time window constraints, algorithms that can effectively handle time-dependent constraints are recommended.

Simulated annealing, tabu search, and variable neighborhood search algorithms are well suited for VRP instances with time windows. These algorithms can incorporate time constraints into their formulations and neighborhood structures to ensure that feasible solutions are generated.

Incorporating time window penalties or soft time window constraints can also be useful to handle situations where strict adherence to time windows is challenging.

5.4 VRP instances with multiple objectives

For VRP instances with multiple objectives, such as minimizing the total distance traveled and maximizing customer satisfaction simultaneously, multi-objective metaheuristic algorithms are appropriate.

Algorithms like the NSGA-II (Non-dominated Sorting Genetic Algorithm II) [14] or SPEA2 (Strength Pareto Evolutionary Algorithm 2) [15, 16] can handle multiple objectives and generate a set of solutions that represent the trade-off between conflicting objectives. These algorithms use dominance-based sorting and selection mechanisms to maintain a diverse set of non-dominated solutions, known as the Pareto front. Decision-makers can then choose the solution from the Pareto front that best aligns with their preferences.

Additionally, the use of preference-based approaches or weighting schemes can help find a single solution that combines multiple objectives into a single objective function [17].

It is important to note that the recommendations provided are general guidelines, and the selection of the most appropriate algorithm for a specific VRP instance should be based on thorough experimentation, considering the problem characteristics, available computational resources, and desired trade-offs between solution quality and computation time.

6. Conclusion

In this chapter, we have explored the application of search algorithms for solving the Vehicle Routing Problem (VRP). We provided an overview of the VRP and its variants, discussing the main challenges and constraints associated with the problem. We then conducted a comparative analysis of various search algorithms used to solve the VRP, including genetic algorithms, simulated annealing, ant colony optimization, and other metaheuristic approaches.

Through our analysis, we found that different search algorithms have their strengths and weaknesses when applied to the VRP. Genetic algorithms excel in handling large-scale instances and complex constraints, while simulated annealing is effective for problems with time window constraints. Ant colony optimization demonstrates robustness in finding good-quality solutions, and other algorithms like tabu search, particle swarm optimization, and iterated local search offer alternative approaches to tackle the VRP.

In the case study section, we provided examples of how search algorithms have been applied to vehicle routing for package delivery, waste collection, and emergency response. We compared the algorithm performance in different scenarios, considering factors such as solution quality, computation time, and constraint satisfaction.

The findings of this chapter highlight the importance of considering problem-specific characteristics, such as problem size, complexity, and constraints, when selecting an algorithm for the VRP. The computational resources available also play a significant role in algorithm performance, and different strategies can be employed to optimize performance within resource constraints.

6.1 Summary of findings

Based on our analysis and case studies, we observed that no single algorithm performs optimally for all types of VRP instances. The choice of algorithm depends on the problem characteristics, including the problem size, complexity, time windows, and multiple objectives. It is essential to evaluate and compare the performance of different algorithms using appropriate metrics to determine the most suitable algorithm for a given VRP instance.

Metaheuristic algorithms, such as genetic algorithms, simulated annealing, and ant colony optimization, demonstrate their effectiveness in solving the VRP. These algorithms provide a good balance between exploration and exploitation, enabling them to handle the complex search space and constraints associated with the VRP.

Additionally, hybridization techniques, parallel computing, and problem decomposition can further enhance the algorithm performance and scalability for large-scale VRP instances.

6.2 Contributions of the study

This chapter contributes to the understanding and analysis of search algorithms for solving the VRP. The key contributions of this study are as follows:

1. Comprehensive overview: We provided a comprehensive overview of the VRP, its variants, and the main challenges and constraints associated with the problem.

2. **Comparative analysis:** We conducted a comparative analysis of different search algorithms, including genetic algorithms, simulated annealing, ant colony optimization, and other metaheuristic approaches, highlighting their strengths and limitations.
3. **Case studies and examples:** We presented case studies and examples of how search algorithms have been applied to real-world scenarios, such as vehicle routing for package delivery, waste collection, and emergency response.
4. **Factors influencing algorithm performance:** We discussed the factors that influence the performance of search algorithms, including problem size, complexity, algorithm parameters, problem-specific constraints, and computational resources.

6.3 Future research directions

While this chapter provides a comprehensive analysis of search algorithms for the VRP, there are several avenues for future research that can further enhance the field:

1. **Development of hybrid algorithms:** Investigate the development of hybrid algorithms that combine the strengths of multiple algorithms to improve solution quality and convergence speed for the VRP.
2. **Incorporation of uncertainty:** Explore the integration of uncertainty into the VRP models and algorithms, considering factors such as traffic congestion, uncertain customer demands, and dynamic time windows.
3. **Integration of machine learning:** Investigate the integration of machine learning techniques, such as deep learning or reinforcement learning, into search algorithms for the VRP. Machine learning can be used to enhance the exploration and exploitation capabilities of algorithms, improve decision-making processes, and adapt the algorithms to evolving problem conditions.
4. **Metaheuristic parameter tuning:** Conduct further research on parameter tuning techniques for metaheuristic algorithms to automatically adjust algorithm parameters based on problem characteristics and instance-specific requirements. This can improve the algorithm's adaptability and performance across different VRP instances.
5. **Multi-objective VRP:** Explore advanced techniques for solving VRP instances with multiple objectives, such as developing new algorithms or improving existing algorithms, to better handle the trade-offs between conflicting objectives. This can include the development of preference-based approaches, interactive algorithms, or Pareto dominance-based methods.
6. **Real-time VRP:** Investigate real-time or dynamic VRP scenarios where new customer requests arrive during the execution of the routing plan. Develop algorithms that can efficiently adapt the existing routes and schedules to incorporate the dynamic changes in real time, considering constraints such as time windows, vehicle capacities, and customer priorities.

7. Benchmark problems and performance metrics: Continuously improve the benchmark problems and performance metrics used for evaluating VRP algorithms. Develop more realistic and challenging problem instances that better capture the complexities of real-world scenarios, and define performance metrics that encompass a wider range of evaluation criteria, including efficiency, robustness, and sustainability.
8. Collaboration and cooperative algorithms: Explore the application of collaboration or cooperative algorithms for the VRP, where multiple vehicles or agents work together to optimize the routing and scheduling process. Investigate communication protocols, coordination mechanisms, and distributed algorithms to solve complex VRP instances more efficiently.


By pursuing these future research directions, we can further advance the field of VRP and develop more effective algorithms to address the challenges and complexities encountered in real-world vehicle routing applications. These advancements will contribute to improving the efficiency, sustainability, and operational effectiveness of transportation and logistics systems.

Author details

Oladimeji Samuel Sowole
African Institute for Mathematical Sciences, Senegal

*Address all correspondence to: oladimeji.s.sowole@aims-senegal.org

IntechOpen

© 2023 The Author(s). Licensee IntechOpen. This chapter is distributed under the terms of the Creative Commons Attribution License (<http://creativecommons.org/licenses/by/3.0>), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited. 

References

- [1] Golden BL, Raghavan S, Wasil EA. The Vehicle Routing Problem: Latest Advances and New Challenges. Vol. 43. Springer Science & Business Media; 2008
- [2] Laporte G. The vehicle routing problem: An overview of exact and approximate algorithms. *European Journal of Operational Research*. 1992;59(3):345358
- [3] Semet F, Toth P, Vigo D. Chapter 2: Classical exact algorithms for the capacitated vehicle routing problem. In: *Vehicle Routing: Problems, Methods, and Applications*. Second ed. SIAM; 2014. pp. 37-57
- [4] Kallehauge B, Larsen J, Madsen OBG, Solomon MM. *Vehicle Routing Problem with Time Windows, Column Generation*. Springer; 2005. pp. 67-98
- [5] Alba E, Dorronsoro B. Solving the vehicle routing problem by using cellular genetic algorithms. In: *Lecture Notes in Computer Science, Proceedings of the Conference on Evolutionary Computation in Combinatorial Optimization, LNCS, Coimbra, Portugal, 5-7 April 2004*. Vol. 3004. Berlin/Heidelberg, Germany: Springer; 2004. pp. 11-20
- [6] Vaira G. *Genetic Algorithm for Vehicle Routing Problem*. Ph.D. Thesis. Vilnius, Lithuania: Vilnius University; 2014
- [7] Yu VF, Susanto H, Jodiawan P, Ho T-W, Lin S-W, Huang Y-T. A simulated annealing algorithm for the vehicle routing problem with parcel lockers. *IEEE Access*. 2022;10:20764-20782. DOI: 10.1109/ACCESS.2022.3152062
- [8] Wang C, Zhao F, Mu D, Sutherland JW. Simulated annealing for a vehicle routing problem with simultaneous pickup-delivery and time windows. In: Prabhu V, Taisch M, Kiritsis D, editors. *Advances in Production Management Systems. Sustainable Production and Service Supply Chains. APMS 2013. IFIP Advances in Information and Communication Technology*. Vol. 415. Berlin, Heidelberg: Springer; 2013. DOI: 10.1007/978-3-642-41263-9_21
- [9] Calvete HI, Galé C, Oliveros MJ. Ant Colony optimization for solving the vehicle routing problem with delivery preferences. In: Engemann KJ, Gil-Lafuente AM, Merigó JM, editors. *Modeling and Simulation in Engineering, Economics, and Management. MS 2012. Lecture Notes in Business Information Processing*. Vol. 115. Berlin, Heidelberg: Springer; 2012. DOI: 10.1007/978-3-642-30433-0_23
- [10] Gurpreetsingh E, Dhir V. *Vehicle Routing Problem by Ant Colony Optimization*. Semantic Scholar; 2014
- [11] Barbarosoglu G, Ozgur D. A tabu search algorithm for the vehicle routing problem. *Computers & Operations Research*. 1999;26(3):255-270. DOI: 10.1016/S0305-0548(98)00047-1
- [12] Marinakis Y, Marinaki M, Migdalas A. Particle swarm optimization for the vehicle routing problem: A survey and a comparative analysis. In: Martí R, Pardalos P, Resende M, editors. *Handbook of Heuristics*. Cham: Springer; 2018. DOI: 10.1007/978-3-319-07124-4_42
- [13] Euchí J, Zidi S, Laouamer L. A hybrid approach to solve the vehicle routing problem with time windows and synchronized visits In-home health care. *Arabian Journal for Science and Engineering*. 2020;45:10637-10652. DOI: 10.1007/s13369-020-04828-5

[14] Deb K, Pratap A, Agarwal S, Meyarivan T. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*. 2002;**6**(2): 182-197. DOI: 10.1109/4235.996017

[15] Amuso VJ, Enslin J. The strength of Pareto evolutionary algorithm 2 (SPEA2) applied to simultaneous multi-mission waveform design. In: 2007 International Waveform Diversity and Design Conference, Pisa, Italy. Vol. 20. 2007. pp. 407-417. DOI: 10.1109/21WDDC.2007.4339452

[16] Long J, Sun Z, Pardalos PM, Hong Y, Zhang S, Li C. A hybrid multi-objective genetic local search algorithm for the prize-collecting vehicle routing problem. *Information Sciences*. 2019;**478**:40-61. DOI: 10.1016/j.ins.2018.11.006

[17] Ait Haddadene SR, Labadie N, Prodhon C. Bicriteria, vehicle routing problem with preferences and timing constraints in home health care services. *Algorithms*. 2019;**12**:152. DOI: 10.3390/a12080152