Route splitting and adaptive mutation in genetic algorithms for the capacitated vehicle routing problem

Shirali Kadyrov[1], Cemil Turan[2]
[1]Department of General Education, Center for Excellence, New Uzbekistan University, Tashkent, Uzbekistan
[2]Department of Computer Science, School of Information Technology and Applied Mathematics, SDU University, Almaty, Kazakhstan

Article Info

Article history:

Received Aug 26, 2024
Revised Sep 29, 2025
Accepted Oct 14, 2025

Keywords:

Chromosome encoding
Genetic algorithm
Heuristic methods
Route optimization
Route splitting
Vehicle routing problem

ABSTRACT

The capacitated vehicle routing problem (CVRP), where vehicle capacity constraints limit the load carried per route for multiple vehicles, is addressed using an optimized genetic algorithm (GA) framework. This work focuses on finding the best configuration of GA by systematically evaluating 12 distinct GA variants, differing in adaptive mutation rates and route-splitting strategies. The framework integrates adaptive mutation rates and novel route-splitting approaches—greedy, dynamic programming (DP), and heuristic—to enhance computational efficiency and solution quality. Experiments on six CVRP instances of varying complexity, encompassing differences in problem size, vehicle capacity, and geographical distribution, demonstrate the heuristic approach's effectiveness. It achieves solutions within 2%–5% of the optimal cost of DP while being 3–4 times faster. Adaptive techniques reduce costs by up to 20% compared to standard GAs and heuristics. The framework's scalability is evident in large-scale instances such as the 200-customer case, where the heuristic method balances cost (414.17) and computation time (0.003 seconds). The developed software is openly available at GitHub, providing a robust tool for addressing practical logistics challenges.

Corresponding Author:

Cemil Turan
Department of Computer Science, School of Information Technology and Applied Mathematics
SDU University
Abylaikhan Street 1/1, Kaskelen, Almaty, Kazakhstan
Email: cemil.turan@sdu.edu.kz

1. INTRODUCTION

Vehicle routing problem (VRP) is a fundamental problem in logistics and supply chain management, where the objective is to determine the most efficient routes for a fleet of vehicles to deliver goods to a set of customers from a central depot [1], [2]. This problem has numerous industrial applications and has been widely studied since its introduction by Dantzig and Ramser [3]. Among the various forms of VRP, the capacitated vehicle routing problem (CVRP) is particularly significant due to the additional constraint of vehicle capacity, which must not be exceeded. The CVRP involves determining a set of routes for vehicles, each with a limited carrying capacity, to service a set of customers with known demands at minimal cost [4], [5].

The CVRP is formally defined on an undirected graph $G = (V, A)$ where V represents the set of vertices (the depot and customers), and A represents the arcs between these vertices. Each arc has an associated cost, typically representing the distance or time required to travel between vertices. The objective is to minimize the total cost while ensuring that each customer's demand is met and that no vehicle exceeds its capacity [6], [7]. Accorsi and Vigo [8] introduced a fast and scalable heuristic for the solution of large-scale CVRPs, focusing on computational time based on the iterated local search paradigm and integrating an effective execution of established acceleration methods for local search engines with many novel algorithmic elements, whose functions and effects are thoroughly examined. Ma *et al.* [9] studied a learning-to-search solver for routing problems, focusing on both optimal solutions and computational efficiency. Overall, the CVRP has been a focal point of research due to its combinatorial complexity and practical importance. Efficient solutions to the CVRP can lead to significant cost savings and operational efficiencies across various industries, including transportation, logistics, and distribution.

The CVRP presents a critical challenge in optimizing routes for multiple vehicles with limited carrying capacity. Despite extensive research, existing approaches, including genetic algorithms (GAs) and hybrid metaheuristics, often fail to balance solution quality and computational efficiency for large-scale instances, suffering from premature convergence, limited solution diversity, and scalability challenges [9], [10]. These limitations hinder practical applicability in complex logistics scenarios, necessitating a novel approach that enhances diversity, scalability, and efficiency. Unlike prior studies, which often emphasize either cost optimization [6] or computational speed [8], this study addresses these gaps by proposing an advanced GA framework that integrates adaptive mutation rates and innovative route-splitting strategies. This approach is designed to achieve a balanced trade-off between cost minimization and computational speed while improving convergence and scalability for large-scale CVRP instances. The objectives are:

a. To design an efficient GA tailored for the CVRP, incorporating optimized crossover and mutation strategies.
b. To evaluate the performance of the proposed algorithm against selected instances and compare it with existing approaches.
c. To investigate the impact of route-splitting techniques and adaptive mutation rates on the efficiency and effectiveness of the GA.

This study contributes to the CVRP literature by optimizing an existing GA framework through the systematic evaluation of 12 configurations that integrate adaptive mutation rates and three innovative route-splitting strategies—greedy, dynamic programming (DP), and heuristic. The proposed approach achieves up to 20% cost reduction and improved scalability over traditional GAs [11], [12], while addressing premature convergence and solution diversity limitations observed in prior work [10]. It also offers a balanced trade-off between cost and computational efficiency for large-scale instances compared to existing heuristics [8], [9].

The remainder of the paper is structured as follows. Section 2 reviews related work, positioning the proposed framework within existing CVRP literature. Section 3 presents the mathematical formulation of the CVRP, defining the problem's constraints and objectives. Section 4 details the methodology, covering the dataset, chromosome encoding, route-splitting strategies, population initialization, crossover, mutation, and experimental setup. Section 5 presents the results, comparing route-splitting strategies, analyzing the impact of adaptive techniques, and evaluating solution performance across CVRP instances. Section 6 discusses the findings, their implications, and limitations. Section 7 concludes with a summary of contributions and directions for future research.

2. RELATED WORKS

Over the decades, numerous approaches have been developed to solve the CVRP, ranging from exact algorithms to heuristics and metaheuristics. Among these, GAs have been widely applied to solve the VRP and its variants [13], including the CVRP. GAs is inspired by the process of natural selection and genetics, by using mechanisms such as selection, crossover, and mutation to evolve solutions over successive generations. Numerous studies have demonstrated the effectiveness of GAs in producing high-quality solutions for CVRP. They have been widely applied to the VRP and its variants. For instance, Baker and Ayechew [11] demonstrated the application of a GA to the basic VRP, highlighting its competitiveness in terms of computing time and solution quality. Their work set the stage for further enhancements and hybrid approaches.

Hybrid metaheuristics have been particularly effective in improving the performance of GAs for solving the CVRP. For example, Nazif and Lee [12] introduced an optimized crossover GA that combines the GA with an optimized crossover mechanism designed by a complete undirected bipartite graph. This approach showed competitive results compared to other heuristics, demonstrating the potential of hybrid strategies in enhancing solution quality and computational efficiency.

Another significant contribution comes from the work on hybrid metaheuristics that integrate various optimization techniques. A study by Berger and Mohamed [14] proposed a hybrid GA for the CVRP, combining elements of local search and GAs to improve performance. Similarly, a chicken swarm optimization with GA and a hybrid chicken swarm optimization with a tabu search algorithm were introduced by Niazy *et al.* [15], [16] for solving the CVRP. This approach integrates the exploration capabilities of chicken swarm optimization with the intensification mechanisms of tabu search, yielding superior results in benchmark tests. They effectively reduced delivery costs while satisfying capacity constraints, highlighting the benefits of combining metaheuristic techniques. Additionally, Cuk *et al.* [17] proposed a hybrid GA and artificial bee colony method to train feedforward multi-layer perceptron, addressing premature convergence, which offers insights for enhancing GA exploration in optimization problems. Similarly, Zivkovic *et al.* [18] developed a hybrid GA and machine learning approach to optimize neuro-fuzzy systems for COVID-19 case prediction, demonstrating the versatility of hybrid GAs in time-series optimization.

Recent advancements also include the integration of fuzzy clustering methods with GAs. Zhu [10] proposed an improved GA based on fuzzy C-means clustering to solve the CVRP. This approach decomposes the large-scale VRP into smaller subproblems, improving the efficiency and robustness of the algorithm. Similarly, Halim *et al.* [19] introduced a GA-based feature selection method for intrusion detection systems, achieving high classification accuracy by optimizing feature subsets, which highlights the potential of GAs in high-dimensional optimization tasks. Additionally, the use of optimized crossover and mutation strategies has been shown to enhance the performance of GAs, as demonstrated in the study by Nazif and Lee [12], which highlighted the importance of maintaining genetic diversity and avoiding premature convergence. Wahab *et al.* [20] proposed an enhanced GA for mobile robot path planning, incorporating novel population initialization and combined genetic operators to improve path quality and convergence, offering relevant techniques for CVRP's initialization and operator design.

Furthermore, ant colony optimization has been effectively applied to the VRP and its variants [21], [22], including the CVRP. Ahmed *et al.* [23] proposed an enhanced ant colony system algorithm that yielded competitive results compared to other heuristics. Similarly, Hendrawan *et al.* [24] addressed the complexity of generating optimal multi-day travel itineraries by framing the problem as a CVRP with a Time Window. Using a hybrid ant colony system and brainstorm optimization (ACS-BSO) algorithm, their study demonstrated the hybrid ACS-BSO algorithm's superiority over conventional in optimizing various travel attributes.

As aforementioned, the CVRP represents a critical challenge in logistics, where optimizing routes for vehicles with limited carrying capacity is paramount. Despite extensive research since its introduction [3], existing approaches often face trade-offs between solution quality and computational efficiency for large scales [9]. While many studies have leveraged GAs and hybrid metaheuristic methods to improve CVRP solutions, these approaches frequently encounter limitations, such as premature convergence, lack of solution diversity, or inefficiency in scaling to larger instances [10]. We aim to address these gaps by introducing a novel GA framework that integrates adaptive mutation rates and advanced route-splitting techniques. By focusing on partial randomness during initialization and heuristic methods for route splitting, we seek to enhance solution diversity and convergence rates. Unlike prior studies, which often emphasize either cost optimization [6] or computational speed [8], this research aims to achieve a balanced trade-off suitable for large-scale CVRP scenarios. Furthermore, the investigation explores the practical implications of adaptive strategies and hybrid initialization on solution quality, highlighting novel contributions to the field of combinatorial optimization by focusing on the impact of chromosome representation and adaptive mutation rate on the performance of GAs in solving the CVRP, building on the foundations laid by previous studies. By addressing these objectives, we aim to contribute to the fields of combinatorial optimization and logistics management, providing a robust and efficient solution for the CVRP. The next section provides a detailed mathematical formulation of the CVRP problem.

3. MATHEMATICAL FORMULATION OF THE CVRP

The CVRP is mathematically formulated to define the optimization problem addressed by the proposed GA. This formulation models the CVRP on a graph, specifying constraints and objectives to ensure efficient route optimization. The following sections describe the graph structure, parameters, decision variables, objective function, and constraints, with detailed explanations of each component's role and the meaning of all variables involved, maintaining an objective perspective.

3.1. Graph structure and parameters

The CVRP is defined on an undirected graph $G=(V, A)$, where V denotes the set of nodes and A denotes the set of arcs. The node set $V=\{0, 1, 2,..., n\}$ comprises the depot, labeled as node 0, and n customers, labeled from 1 to n, where n represents the total number of customers. Each node $i \in V$ represents either the depot $(i=0)$ or a customer $(i=1, 2,..., n)$, and each node $j \in V$ similarly represents another node in the graph. The arcs in A represent possible routes between pairs of nodes, including between the depot and customers, with each arc $(i, j) \in A$ associated with a cost c_{ij}, typically representing the distance or travel time between nodes i and j. The set K represents the fleet of identical vehicles available to serve customers, each with a fixed capacity Q, which limits the total demand a vehicle can carry. Each customer $i \in \{1, 2,..., n\}$ has a demand q_i, indicating the quantity of goods to be delivered. These parameters establish the problem's structure, ensuring that routes account for travel costs, vehicle availability, capacity limits, and customer demands.

3.2. Decision variables

The formulation employs two primary decision variables to represent routing and load decisions. The binary variable x_{ijk} indicates whether vehicle $k \in K$ travels directly from node i to node j along arc $(i, j) \in A$: it equals 1 if the arc is used by vehicle k, and 0 otherwise. This variable determines the structure of the routes assigned to each vehicle. The variable y_{ik} represents the cumulative load carried by vehicle k after visiting node i, tracking the total demand serviced along its route to ensure compliance with the capacity constraint Q. These variables facilitate the optimization of customer assignments to vehicle routes and the management of vehicle loads.

3.3. Objective function

The objective of the CVRP is to minimize the total cost of all vehicle routes, expressed as (1):

$$minimize \sum_{k\in K} \sum_{(i,j)\in A} c_{ij}x_{ijk} \quad (1)$$

This function computes the sum of the costs cij for all arcs (i, j) traversed by any vehicle k, where $x_{ijk}=1$. Minimizing this sum identifies the most cost-efficient set of routes that satisfies all constraints, aligning with practical logistics goals such as reducing fuel consumption or travel time.

3.4. Constraints

The CVRP includes several constraints to ensure feasible and practical solutions:

a. Customer visit constraint

Each customer must be visited exactly once by one vehicle, enforced by:

$$\sum_{k\in K} \sum_{j\in V, j\neq i} x_{ijk} = 1 , \forall i \in \{1,2, \dots , n\} \quad (2)$$

This constraint ensures that for each customer i (from 1 to n), exactly one vehicle k travels to it from some node j, guaranteeing that every customer's demand is met without duplication.

b. Load constraint and sub-tour elimination

To manage vehicle loads and prevent sub-tours (cycles excluding the depot), the following constraint is applied:

$$y_{jk} \geq y_{ik} + q_j - Q(1 - x_{ijk}),\ \forall k \in K, (i,j) \in A, i \neq j, j \neq 0 \quad (3)$$

This ensures that if vehicle k travels from node i to node j $(x_{ijk}=1)$, the load after visiting j (y_{jk}) is at least the load after visiting i (y_{ik}) plus the demand at j (q_j). The term $Q(1\text{-}x_{ijk})$ deactivates the constraint when the arc is not used (x_{ijk}=0), and it prevents sub-tours by enforcing consistent load accumulation tied to the depot.

c. Depot load constraint

At the depot (node 0), each vehicle starts with no load:

$$y_{0k} = 0, \forall k \in K \quad (4)$$

This initializes the load of each vehicle k to zero upon departing from the depot, ensuring accurate tracking of load accumulation as customers are visited.

d. Vehicle capacity constraint

The load carried by each vehicle must not exceed its capacity and must remain non-negative:

$$0 \leq y_{ik} \leq Q, \forall k \in K, i \in V \quad (5)$$

This ensures that the cumulative load y_{ik} for vehicle k at any node i (including the depot and customers) respects the vehicle's capacity Q, preventing overloading while allowing feasible delivery schedules.

e. Binary decision variable constraint

The routing decisions are binary:

$$x_{ijk} \in \{0,1\},\ \forall k \in K, (i,j) \in A \tag{6}$$

This mathematical formulation establishes a rigorous framework for the CVRP, capturing the essential elements of route optimization under capacity constraints. The parameters define the problem's structure, the decision variables model routing and load decisions, the objective function targets cost minimization, and the constraints ensure feasible solutions. The proposed GA leverages this formulation to generate efficient routes, addressing the combinatorial complexity of the CVRP while optimizing for both solution quality and computational efficiency.

4. METHOD

The CVRP is addressed through a GA framework comprising data preparation, chromosome encoding, population initialization, crossover, mutation, and route-splitting strategies. This section details the methodology, starting with the dataset of CVRP instances, followed by chromosome encoding and three route-splitting approaches (greedy, DP, and heuristic), population initialization with partial randomness and heuristics, the alternating edges crossover (AEX) operator, adaptive mutation rate control, and the experimental setup, with implementation details available in a GitHub repository [25].

4.1. Dataset

In the context of the CVRP, an *instance* refers to a specific problem configuration defined by a set of customers, their demands, vehicle capacities, and geographical locations, representing a unique test case to evaluate algorithm performance. For this study, six instances of the CVRP were generated to evaluate the proposed GA thoroughly. These instances were selected to encompass a range of problem sizes, vehicle capacities, and geographical distributions. Table 1 summarizes the characteristics of these CVRP instances.

Table 1. CVRP instances

CVRP instance	Number of customers (n)	Number of vehicles (K)	Vehicle capacity (Q)	Total demand $\sum q_i$
CVRP20	20	1	100	87
CVRP50	50	3	100	258
CVRP75	75	4	100	357
CVRP100	100	5	100	467
CVRP150	150	8	100	752
CVRP200	200	11	100	1067

Figure 1 plots the instances to visually represent their distribution and diversity across problem dimensions, such as the number of customers, number of vehicles, vehicle capacity, and total demand. This graphical representation further underscores the comprehensive nature of our evaluation approach.

4.2. Chromosome encoding and route splitting approaches

For a CVRP problem with n customers, any permutation of $\{1,2,\dots,n\}$ will represent a chromosome. The permutation representation approach is one of the most common approaches, see e.g., [12], [26]-[29]. However, one still needs to split the given chromosome into routes with different vehicles to represent a solution. This means inserting 2K zeros into the chromosome sequence to represent each of the K vehicles leaving and returning to the depot. Mathematically, if we denote a chromosome as;

$$c = (c_1, c_2, \dots, c_n)$$

where c_i represents customer i, then adding $2K$ zeros can be expressed as:

$$c' = (0, c_1, c_2, \dots, c_k, 0, c_{k+1}, \dots, c_n, 0)$$

where each zero indicates a depot visit. The challenge lies in determining where to place these zeros optimally to form valid routes while adhering to vehicle capacity constraints. In this paper, we experiment with three different approaches for route splitting.

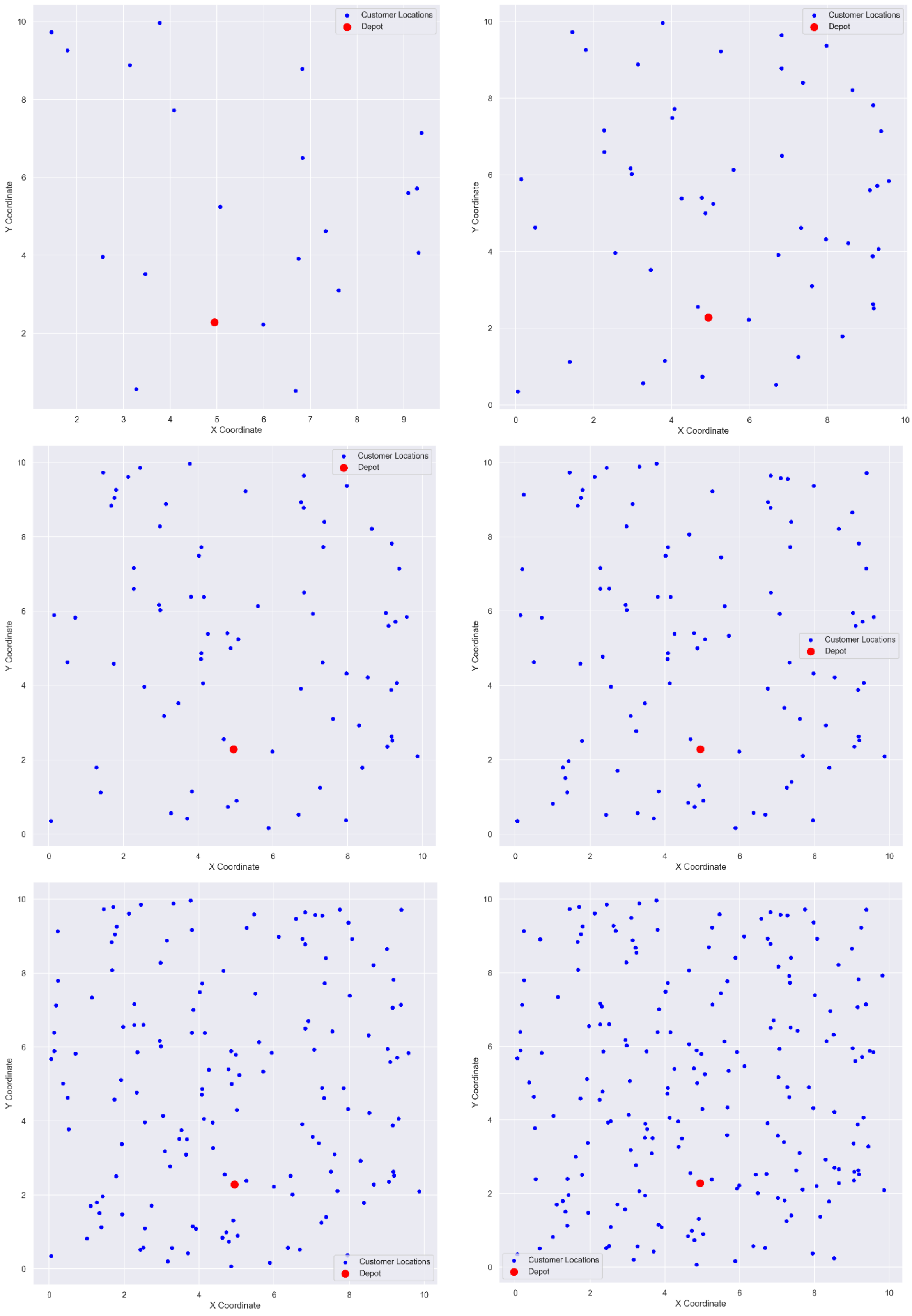


Figure 1. Scatter plots of six CVRP instances

4.2.1. Greedy approach

The idea in the greedy approach is to partition a chromosome representing a solution to the CVRP into feasible routes. Starting with an initial route that begins and ends at the depot (customer 0), the function iterates through each customer in the chromosome, attempting to add them to the current route without exceeding the vehicle's capacity constraint. If adding a customer surpasses this capacity, the current route is finalized by appending the depot and stored in the routes list, and a new route begins with the depot and the current customer. This process continues until all customers have been assigned to routes, resulting in a list of routes where each route starts and ends at the depot and respects the vehicle capacity limitations. The pseudocode for this approach is shown in Algorithm 1.

Algorithm 1. Greedy route splitting for CVRP
```
Input:
    individual ← a permutation of customers (chromosome)
    demand ← a list of customer demands
    vehicle_capacity ← the capacity of each vehicle

Output:
    routes ← a list of routes where each route starts and ends at the depot

1. Initialize:
    routes ← an empty list
    route ← [0]  // Start route with depot
    load ← 0  // Current load on vehicle

2. For each customer in individual do:
    a. If load + demand[customer] > vehicle_capacity then:
        i.  Append route + [0] to routes  // Finalize current route
        ii. Start a new route: route ← [0, customer]
        iii. Reset load: load ← demand[customer]
    b. Else:
        i.  Append customer to route
        ii. Update load: load ← load + demand[customer]

3. After all customers are processed:
    Append route + [0] to routes  // Finalize the last route

4. Return routes
```

4.2.2. Dynamic programming approach

The algorithm given in Algorithm 2 employs a DP approach to efficiently partition a chromosome (representing a solution to the CVRP) into optimal routes that adhere to vehicle capacity constraints.

Algorithm 2. DP-based route splitting for CVRP
```
Input:
    individual ← a permutation of customers (chromosome)
    cost_matrix ← a matrix of costs between customers
    demand ← a list of customer demands
    vehicle_capacity ← the capacity of each vehicle

Output:
    final_routes ← a list of routes where each route starts and ends at the depot
    best_cost ← the minimum cost of the best solution found

1. Initialize:
    n ← length of individual  // Number of customers
    dp ← array of size n+1 with values set to infinity
    dp[0] ← 0  // Starting point, no cost for 0 customers
    split_point ← array of size n+1 with values set to -1

2. For i=1 to n do:
    a. Initialize load ← 0  // Current load on the vehicle
    b. For j=i down to 1 do:
        i.  Add demand[individual[j-1]] to load
        ii. If load > vehicle_capacity then break  // Exceeds capacity
        iii. Calculate cost as:
            cost=dp[j-1] + cost_matrix[0][individual[j-1]] +
                   sum(cost_matrix[individual[k]][individual[k+1]] for k in
range(j-1, i-1)) +
                   cost_matrix[individual[i-1]][0]
        iv. If cost < dp[i] then:
            Update dp[i] ← cost
            Update split_point[i] ← j-1

3. Backtrack to determine the routes:
    i ← n  // Start from the last customer
    routes ← empty list

    While i > 0 do:
        j ← split_point[i]
        Append [0] + individual[j:i] + [0] to routes
        i ← j

4. Reverse the routes list to get the correct order

5. Return final_routes, dp[n]
```

It initializes an array dp to store minimum costs for subproblems and iteratively evaluates potential partitions of the chromosome based on vehicle load capacities. The algorithm efficiently determines the optimal route configuration using a nested loop structure by computing cumulative costs for routes starting from the depot, visiting customers, and returning. After computing the minimal costs, it uses backtracking through split points to systematically construct each route, ensuring all routes begin and end at the depot while respecting capacity limits. This approach ensures not only the identification of the optimal partitioning of the chromosome into routes but also provides the total cost of the solution, making it well-suited for solving CVRP instances where minimizing travel costs and adhering to vehicle capacities are critical objectives.

4.2.3. Heuristic approach

DP runs slowly while finding the route splitting with minimal cost. As a third approach, we implement a GA to find near-optimal route splitting. It begins by initializing an initial solution of routes based on the given individual, where each route respects the vehicle capacity constraints. The function iteratively refines these routes over 50 iterations, employing a mutation operation with a 10% probability of exploring new configurations. During each iteration, it evaluates the cost of mutated routes using a cost matrix that defines travel expenses between customers and the depot. By continually updating to lower-cost solutions found through mutation, the function aims to converge on an optimal set of routes that start and end at the depot while minimizing total travel costs. This approach suits CVRP scenarios where efficient vehicle routing and cost reduction are crucial objectives. The pseudocode for this approach is shown in Algorithm 3.

Algorithm 3. GA-inspired route splitting for CVRP

```
 Input:
     individual ← a permutation of customers (chromosome)
     cost_matrix ← a matrix of costs between customers
     demand ← a list of customer demands
     vehicle_capacity ← the capacity of each vehicle

 Output:
     final_routes ← a list of routes where each route starts and ends at the depot
     best_cost ← the cost of the best solution found

 1. Initialize:
     n ← number of customers (len(demand) - 1)
     chromosome ← individual[:]  // Copy of the chromosome
     best_routes ← initial_solution(chromosome)  // Generate an initial solution
     best_cost ← evaluate_solution(best_routes, cost_matrix)  // Evaluate initial solution

 2. Define initial_solution(chromosome):
     routes ← empty list
     route ← empty list
     load ← 0  // Current load on vehicle

     For each gene in chromosome do:
         If load + demand[gene] <= vehicle_capacity:
             Append gene to route
             Update load: load ← load + demand[gene]
         Else:
             Append route to routes
             Start new route with gene
             Reset load: load ← demand[gene]

     If route is not empty:
         Append route to routes
```

```
    Return routes

3. Define mutate(routes):
    mutated_routes ← copy of routes

    If random chance < 0.1 (10% chance):
        idx1 ← random index from 0 to len(mutated_routes)-1
        idx2 ← random index from 0 to len(mutated_routes)-1

        If idx1 ≠ idx2 and both routes have more than one customer:
            gene1 ← random index in mutated_routes[idx1]
            gene2 ← random index in mutated_routes[idx2]
            Swap mutated_routes[idx1][gene1] and mutated_routes[idx2][gene2]

    Return mutated_routes

4. For each iteration (e.g., 50 iterations):
    mutated_routes ← mutate(best_routes)
    mutated_cost ← evaluate_solution(mutated_routes, cost_matrix)

    If mutated_cost < best_cost:
        Update best_routes ← mutated_routes
        Update best_cost ← mutated_cost

5. Finalize the routes by adding depot at the beginning and end of each route:
    final_routes ← [[0] + route + [0] for each route in best_routes]

6. Return final_routes, best_cost
```

4.3. Partial randomness with heuristics in population initialization

To investigate the efficacy of partial randomness with heuristics in chromosome encoding for the CVRP, this study systematically compares and evaluates different initialization strategies within a GA framework. The research focuses on integrating randomness and heuristic methods, specifically the nearest neighbour and savings algorithms, to generate the initial population of chromosomes. A subset of the population is randomly generated to promote diversity, while another subset utilizes heuristic methods to leverage local optimization based on distance savings and nearest customer selection. This hybrid approach is designed to capitalize on the strengths of both randomness and structured heuristic algorithms, aiming to enhance solution quality, increase solution diversity, and expedite convergence towards optimal or near-optimal solutions. The experiments are conducted using standard CVRP instances, with comprehensive performance metrics employed to assess the impact on solution fitness, diversity, and convergence speed across multiple experimental trials. Statistical analysis is employed to validate findings and draw meaningful conclusions regarding the effectiveness of the hybrid initialization strategy in improving GA performance for CVRP.

4.4. Alternating edges crossover

Eight different crossover approaches were introduced in [13] and compared for their performance. This study adopts the AEX operator due to its demonstrated superiority over other methods. AEX interprets a chromosome as a directed cycle of arcs. Forming a child cycle involves alternately selecting arcs from each parent, incorporating additional random selections if necessary to prevent infeasibility.
For example, consider the following two parent chromosomes:

Parent 1: (2 7 4 9 1 5 3 6 8).

Parent 2: (1 9 6 8 3 2 4 7 5).

The procedure begins by choosing the first arc from Parent 1, specifically the arc 2 → 7. This initializes the child chromosome as:

$c = (2\ 7 * * * * * * *)$.

Next, the algorithm selects the arc from Parent 2 that follows vertex 7, which is 7 → 5. The child chromosome now appears as:

$c = (2\ 7\ 5\ * * * * * *)$.

Continuing this process, the next arc selected is from Parent 1, following vertex 5, which is 5 → 3:

$c = (2\ 7\ 5\ 3\ * * * * *)$

When the algorithm attempts to select the next arc from Parent 2 for vertex 3, it encounters an infeasibility issue because this would prematurely complete the cycle if we add 2. To resolve this, a random choice from the remaining unvisited vertices 1,4,6,8,9 is made. For example, we may choose vertex 4, resulting in the child chromosome:

$c = (2\ 7\ 5\ 3\ 4\ * * * *)$

The procedure then resumes as normal, alternating between parents. The next arcs chosen are $4 \rightarrow 9$ from Parent 1 and $9 \rightarrow 6$ from Parent 2, then $6 \rightarrow 8$ from Parent 1 leading to:

$c = (2\ 7\ 5\ 3\ 4\ 9\ 6\ 8\ *)$

Then, we again encounter an infeasible move $8 \rightarrow 3$, we randomly choose from the unvisited vertices. In this case, we are only left with 1. Hence, the AEX crossover provides the child chromosome:

$c = (2\ 7\ 5\ 3\ 4\ 9\ 6\ 8\ 1)$

This method ensures the child inherits traits from both parents while maintaining feasibility through random selections when necessary.

4.5. Adaptive mutation

The methodology employed in this study focuses on optimizing the performance of a GA for solving the CVRP through adaptive mutation rate control. Initially, a moderate mutation rate of 50% is set, which dynamically adjusts based on the algorithm's performance over monitored iterations. Specifically, the algorithm continuously evaluates fitness improvement over 500 iterations. If minimal or no improvement is detected, the mutation rate decreases incrementally by 5 percentage points, ensuring it does not fall below a minimum threshold of 5%. This adaptive approach aims to enhance exploration during the early stages of the search process, promoting diversity in solutions while gradually shifting towards exploitation in later stages to refine and converge towards optimal or near-optimal solutions.

5. SIMULATION RESULTS

5.1. Experimental setup

There are six distinct CVRP instances, three different route-splitting methods, and two types of mutation strategies—adaptive and standard. For population initialization, we use either random or partially random initialization as described earlier. Testing all possibilities would result in 72 different experiments, which is too many. Instead, we use CVRP50 to determine the best initialization and mutation methods. We then apply these optimal choices to the remaining experiments. Whenever possible, we report cost and time as performance metrics. The experiments were conducted on a MacBook Air equipped with an Apple M2 chip, 8 GB of RAM, running macOS Sonoma 14.5. The implementation used Python 3.12.0, with key libraries including random, numpy, networkx, sklearn, and cluster.

5.2. Comparison of route-splitting strategies for chromosome-based route generation

The GA generates a chromosome as a sequence of customer nodes for multiple vehicles, excluding the depot, requiring route-splitting strategies to insert depot nodes and satisfy capacity constraints of the CVRP. Three route-splitting strategies—greedy, DP, and heuristic—are evaluated on a fixed chromosome to compare their performance in terms of route cost and splitting time across various CVRP instances. Table 2 presents the average cost and splitting time for these route-splitting strategies, derived from 100 experimental runs. Each column represents the cost and time associated with the greedy, DP, and heuristic strategies across different CVRP instances. Results indicate that the DP approach typically yields the lowest route cost, the greedy approach minimizes splitting time, and the heuristic approach achieves a balance between cost and computational efficiency.

Table 2. Cost and time (seconds) for various route-splitting strategies

	Greedy		DP		Heuristic	
Problem	Cost	Time	Cost	Time	Cost	Time
CVRP20	98	8e-6	98	5e-4	98	3e-4
CVRP50	259	2e-5	252	2e-3	255	7e-4
CVRP75	400	3e-5	390	4e-3	394	1e-3
CVRP100	541	4e-5	527	6e-3	536	1e-3
CVRP150	804	6e-5	782	9e-3	798	2e-3
CVRP200	2055	6e-5	1019	1e-2	1051	3e-3

5.3. Performance of three route splitting strategies on a single chromosome

Table 2 compares the average cost and average time for three different route-splitting approaches derived from chromosomes. Each experiment is carried out 100 times, and the average values are presented. Each column represents the cost and time associated with different route-splitting strategies across various CVRP instances. The values show that the DP approach generally results in the lowest cost, while the Greedy approach tends to be the fastest, and the heuristic approach balances time and cost.

5.4. Effect of partial randomness and adaptive crossover

To understand the effect of partial randomness in initialization and adaptive mutation, we conducted experiments on the CVRP50 instance. The results are presented in Table 3. For the non-adaptive mutation, the mutation rate was fixed at 10%.

Table 3. Impact of partial randomness in initialization and adaptive mutation

Partial	Adaptive	Greedy 20k generations		DP 1k generations		Heuristic 1k generations	
		Cost	Time	Cost	Time	Cost	Time
Yes	Yes	**76**	**51**	**98**	**237**	**98**	**67**
Yes	No	98	52	99	235	110	67
No	Yes	83	52	139	227	153	67
No	No	123	52	124	231	138	67

Table 3 shows that partial randomness in population initialization and adaptive mutation significantly impacts the performance of all models. Regarding experiment running time, the DP approach is the slowest. For solving CVRP with 50 customers, the heuristic approach is approximately 3.5 times faster than DP. The greedy approach is the fastest, performing 26 times faster than the heuristic approach and 92 times faster than DP. Figures 2 to 4 show the fitness progress over 1000 generations for various route-splitting approaches. Note that fitness is calculated as the inverse of the total cost.

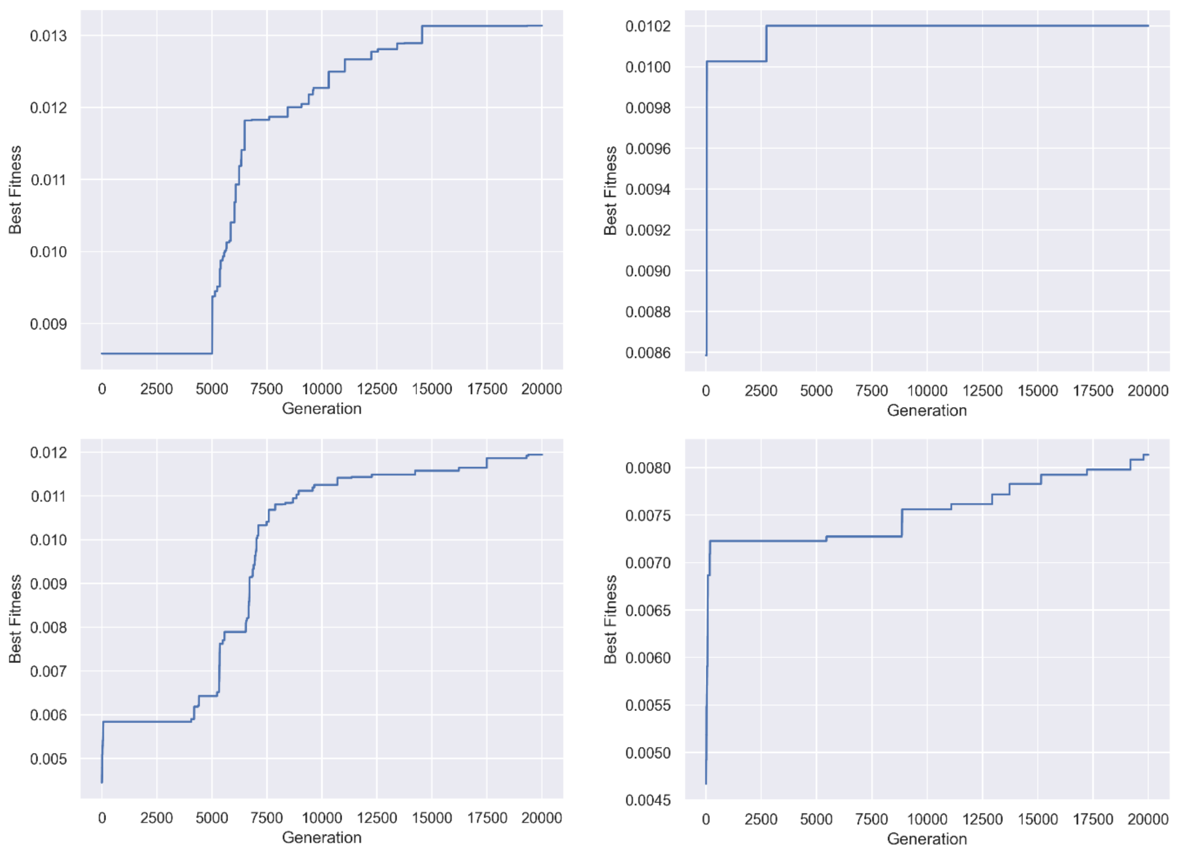


Figure 2. Fitness progress over generations for greedy route splitting

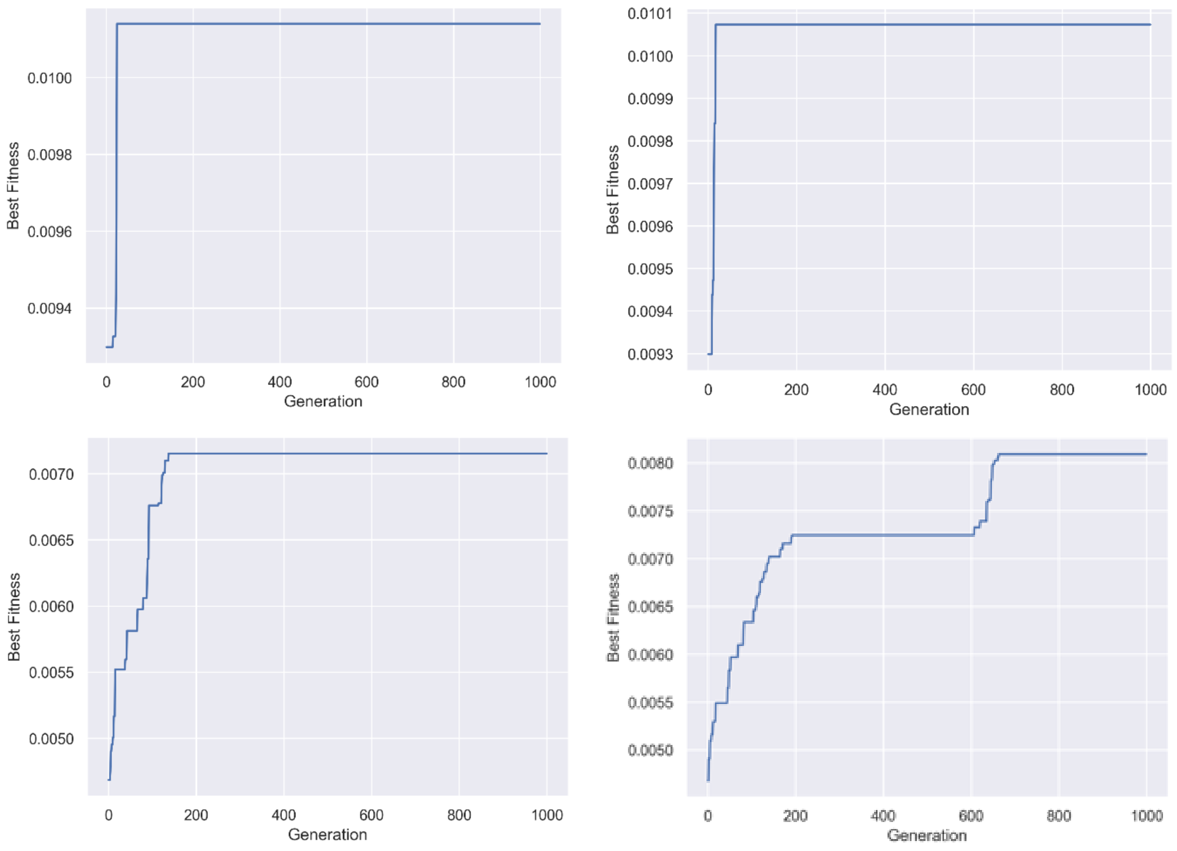


Figure 3. Fitness progress over generations for DP in route splitting

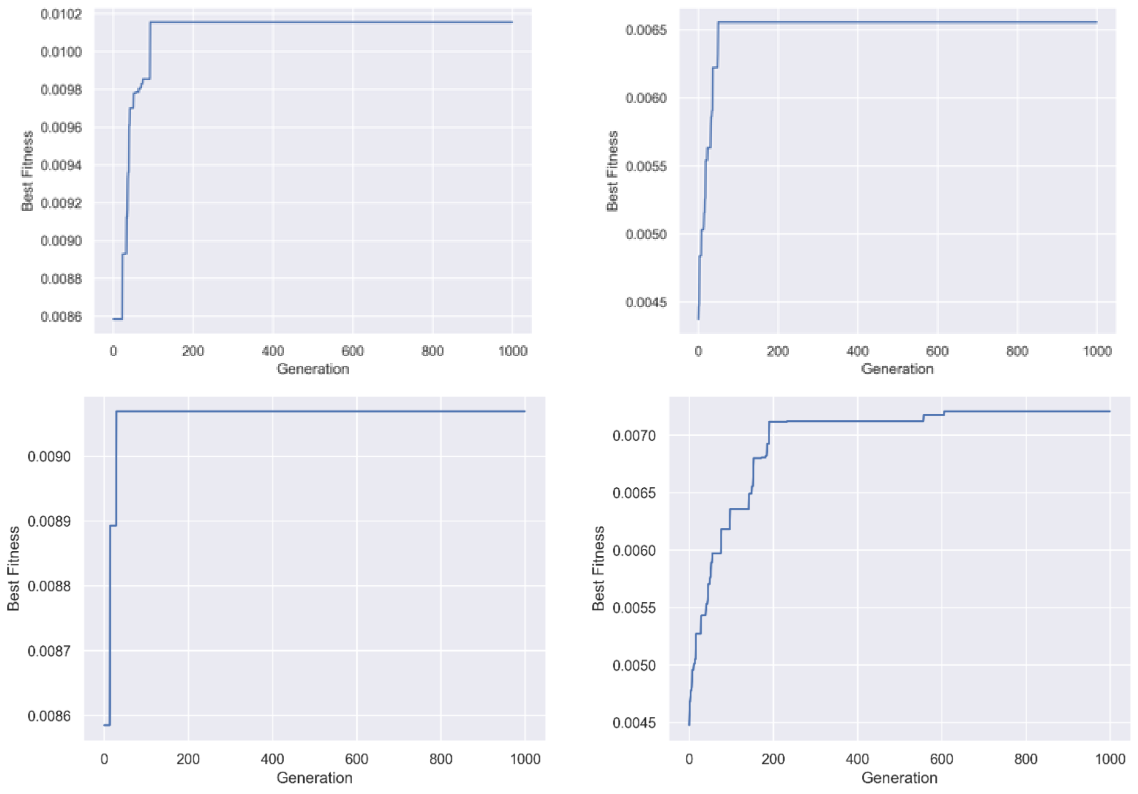


Figure 4. Fitness progress over generations for heuristic route splitting

5.5. Solution performance of proposed methods on various capacitated vehicle routing problem instances

As demonstrated in the previous subsection, partial randomness in initialization and adaptive mutation significantly enhance performance; we therefore implement both partial randomness and adaptive mutation for the remaining experiments. Table 4 summarizes solution costs over 2,000 generations across six CVRP instance sizes for the Greedy, DP, and heuristic route-splitting models. The heuristic achieves the lowest cost in three of six cases (and ties at CVRP50), remaining competitive elsewhere.

Table 4. Solution cost for different models in 2000 generations

Instance	Greedy	DP	Heuristic
CVRP20	**43.05**	48.68	46.15
CVRP50	95.78	**91.60**	**91.60**
CVRP75	143.16	146.05	**132.27**
CVRP100	210.31	196.75	**179.46**
CVRP150	332.11	**311.43**	312.74
CVRP200	422.26	**385.99**	414.17

Next, we present the best solutions obtained by our proposed methods. Specifically, since the heuristic route-splitting approach balances cost optimization and running time, we report the best solutions according to this approach in Table 5.

Table 5. Cost for the best solutions according to heuristic route-splitting

CVPR20	CVPR50	CVPR75	CVPR100	CVPR150	CVPR200
39.26	76.98	117.73	174.26	268.39	410.97

The results, depicted in Figure 5 (in Appendix), from a to f, illustrate the effectiveness of our method across different CVRP instances. Figure 5(a) shows the heuristic route splitting for CVPR20 over 10k generations, highlighting the balance between cost and computational efficiency. Similarly, Figures 5(b)-(f) provide visual representations of the best solutions for CVPR50, CVPR75, CVPR100, CVPR150, and CVPR200 over varying generations, demonstrating consistent improvements in route optimization.

6. DISCUSSION

The results in Tables 2–5 and Figures 2–5 demonstrate the trade-offs and performance characteristics of the proposed methods across various CVRP instances. DP achieves the lowest costs, reducing the total cost by up to 5%–7% compared to the heuristic approach for larger problems like CVRP200. However, DP requires significantly more computational time, often 3–4 times longer than the heuristic approach and up to 92 times longer than the Greedy method. While the Greedy approach is the fastest—requiring as little as 6e-5 seconds for CVRP200—it produces solutions that are up to 10% costlier than those from the DP method. The heuristic approach consistently balances performance, achieving solutions within 2%–5% of DP's optimal cost, while being 3–4 times faster. The effect of adaptive techniques is evident in Table 3. For example, adaptive mutation and partial randomness improved solution quality by reducing costs by up to 20% compared to methods without these enhancements. For CVRP50, costs dropped from 123 to 76 when both adaptive mutation and randomness were applied, illustrating their effectiveness. Despite these improvements, computational time remained nearly constant, underscoring the efficiency of these enhancements. Fitness progress over generations, shown in Figures 2–4, reveals distinct convergence patterns. DP converges steadily but slowly, while the Greedy approach converges rapidly, albeit with higher final costs. The heuristic method demonstrates robust and balanced convergence, combining fast progress with competitive final costs.

The scalability of the methods is evaluated in Table 4, showing that DP struggles with larger instances. For CVRP200, the cost of 385.99 is the lowest, but it requires over 0.01 seconds per iteration, compared to the heuristic's 414.17 in just 0.003 seconds. Meanwhile, the Greedy method achieves a cost of 422.26, emphasizing its suitability for real-time applications where speed is critical. The results in Table 5 and Figure 5 highlight the heuristic method's practical value, delivering consistently competitive solutions. For example, in CVRP150, the best solution cost achieved was 268.39, demonstrating a balanced trade-off between cost and computational time. Across all CVRP instances, the heuristic method proves to be an effective compromise, providing scalable and efficient performance for real-world routing problems. These findings emphasize the importance of selecting an approach based on the specific trade-offs between cost, time, and problem complexity.

The findings, while robust, are subject to certain limitations that contextualize their applicability. The DP approach, despite achieving the lowest costs, exhibits high computational intensity, making it less practical for real-time or large-scale CVRP applications. The heuristic approach, although balanced, requires further refinement to enhance its performance on extremely large instances, such as those with hundreds of customers or complex constraints. Additionally, the evaluation is limited to single-depot CVRP instances without time windows, which may restrict the generalizability to more complex variants of the problem. Furthermore, the ablation study could be expanded to provide a more granular analysis of how individual components—such as adaptive mutation alone or route-splitting alone—contribute to overall performance, which would help isolate the impact of each component more clearly.

7. CONCLUSION

This study evaluates 12 configurations of an existing GA framework for solving the CVRP, integrating adaptive mutation rates and innovative route-splitting strategies. The focus is on optimizing the algorithm's configuration rather than proposing a new algorithm. The comparative analysis of greedy, DP, and heuristic route-splitting methods across diverse CVRP instances demonstrates the heuristic method's ability to balance cost minimization and computational efficiency, offering a scalable solution for practical logistics challenges.

To extend the applicability of the proposed framework, future research may explore the integration of multi-depot and time-window constraints to address more complex CVRP variants. Additionally, incorporating machine learning models for dynamic mutation rate adjustment, hybrid metaheuristics2 combining genetic acceleration techniques for DP, particularly for efficient route splitting, could significantly improve computational performance. In parallel, incorporating other optimization techniques and developing a novel crossover operator tailored to CVRP could further enhance solution quality and convergence rates, building on existing crossover strategies.

FUNDING INFORMATION

This research is funded by the Science Committee of the Ministry of Science and Higher Education of the Republic of Kazakhstan (Grant No. AP19675614).

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Shirali Kadyrov		✓	✓			✓		✓	✓	✓	✓	✓		
Cemil Turan	✓			✓	✓		✓			✓	✓		✓	✓

C	:	**C**onceptualization	I	:	**I**nvestigation	Vi	:	**Vi**sualization
M	:	**M**ethodology	R	:	**R**esources	Su	:	**Su**pervision
So	:	**So**ftware	D	:	**D**ata Curation	P	:	**P**roject administration
Va	:	**Va**lidation	O	:	Writing - **O**riginal Draft	Fu	:	**Fu**nding acquisition
Fo	:	**Fo**rmal analysis	E	:	Writing - Review & **E**diting			

CONFLICT OF INTEREST STATEMENT

Authors state no conflict of interest.

DATA AVAILABILITY

All experimental data consist of synthetic CVRP instances generated by our code, which—together with reproduction scripts—is publicly available in the repository cited as [25].

REFERENCES

[1] N. Pillay and R. Qu, *Vehicle Routing Problems*, Hyper-Heuristics: Theory and Applications, Springer, Cham, pp. 51–60, 2018, doi: 10.1007/978-3-319-96514-7_7.

[2] A. Yernar and C. Turan, "Recent developments in vehicle routing problem under time uncertainty: a comprehensive review," *Bulletin of Electrical Engineering and Informatics*, vol. 14, no. 2, pp. 1263–1275, 2025, doi: 10.11591/eei.v14i2.8636.

[3] G. Dantzig and J. Ramser, "The truck dispatching problem," *Management Science*, vol. 6, no. 1, pp. 80-91, 1959, doi: 10.1287/mnsc.6.1.80.

[4] S. Kadyrov, A. Azamov, Y. Abdumajitov, and C. Turan, "Deep reinforcement learning for dynamic vehicle routing with demand and traffic uncertainty," *Operations Research Perspectives*, vol. 15, pp. 1-12, 2025, doi: 10.1016/j.orp.2025.100351.

[5] V. H. S. Pham, V. N. Nguyen, and N. T. N. Dang, "Novel hybrid swarm intelligence algorithm for solving the capacitated vehicle routing problem efficiently," *Evolutionary Intelligence*, vol. 18, no. 3, 2025, doi: 10.1007/s12065-025-01048-4.

[6] J. Fitzpatrick, D. Ajwani, and P. Carroll, "A scalable learning approach for the capacitated vehicle routing problem," *Computers & Operations Research*, vol. 171, pp. 1-15, 2024, doi: 10.1016/j.cor.2024.106787.

[7] E. Yuliza, F. M. Puspita, and S. S. Supadi, " Heuristic approach for robust counterpart open capacitated vehicle routing problem with time windows," *Science and Technology Indonesia*, vol. 6, no. 2, pp. 53-57, 2021, doi: 10.26554/STI.2021.6.2.53-57.

[8] L. Accorsi and D. Vigo, "A fast and scalable heuristic for the solution of large-scale capacitated vehicle routing problems," *Transportation Science*, vol. 55, no. 4, pp. 832-856, 2021, doi: 10.1287/trsc.2021.1059.

[9] Y. Ma, Z. Cao, and Y. M. Chee, "Learning to search feasible and infeasible regions of routing problems with flexible neural k-opt," *arXiv*, 2023, doi: 10.48550/arXiv.2310.18264.

[10] J. Zhu, "Solving capacitated vehicle routing problem by an improved genetic algorithm with fuzzy c-means clustering," *Scientific Programming*, vol. 2022, no. 1, pp. 1-8, 2022, doi: 10.1155/2022/8514660.

[11] M. B. Baker and M. A. Ayechew, "A genetic algorithm for the vehicle routing problem," *Computers & Operations Research*, vol. 30, no. 5, pp. 787-800, 2003, doi: 10.1016/S0305-0548(02)00051-5.

[12] H. Nazif and L. S. Lee, "Optimised crossover genetic algorithm for capacitated vehicle routing problem," *Applied Mathematical Modelling*, vol. 36, no. 5, pp. 2110-2117, 2012, doi: 10.1016/j.apm.2011.08.010.

[13] Z. H. Ahmed, N. Al-Otaibi, A. Al-Tameem, and A. K. J. Saudagar, "Genetic Crossover Operators for the Capacitated Vehicle Routing Problem," *Computers, Materials & Continua*, vol. 74, no. 1, pp. 1575–1605, 2023, doi: 10.32604/cmc.2023.031325.

[14] J. Berger and B. Mohamed, "A parallel hybrid genetic algorithm for the vehicle routing problem with time windows," *Computers & Operations Research*, vol. 31, no. 12, pp. 2037-2053, 2004, doi: 10.1016/S0305-0548(03)00163-1.

[15] N. Niazy, A. El-Sawy, and M. Gadallah, "Solving capacitated vehicle routing problem using chicken swarm optimization with genetic algorithm," *International Journal of Intelligent Engineering and Systems*, vol. 13, no. 5, pp. 502-513, 2020, doi: 10.22266/ijies2020.1031.44.

[16] N. Niazy, A. El-Sawy, and M. Gadallah, "A hybrid chicken swarm optimization with tabu search algorithm for solving capacitated vehicle routing problem," *International Journal of Intelligent Engineering and Systems*, vol. 13, no. 4, pp. 237-247, 2020, doi: 10.22266/ijies2020.0831.21.

[17] A. Cuk *et al.*, *Feedforward multi-layer perceptron training by hybridized method between genetic algorithm and artificial bee colony*, Data Science and Data Analytics: Opportunities and Challenges, 1st edition, 2021.

[18] M. Zivkovic *et al.*, "Hybrid genetic algorithm and machine learning method for covid-19 cases prediction," in *Proceedings of International Conference on Sustainable Expert Systems*, Springer, pp. 169–184, 2021, doi: 10.1007/978-981-33-4355-9_14.

[19] Z. Halim *et al.*, "An effective genetic algorithm-based feature selection method for intrusion detection systems," *Computers & Security*, vol. 110, pp. 102448, 2021, doi: 10.1016/j.cose.2021.102448.

[20] M. N. Ab Wahab *et al.*, "Improved genetic algorithm for mobile robot path planning in static environments," *Expert Systems with Applications*, vol. 249, part. C, 2024, doi: 10.1016/j.eswa.2024.123762.

[21] Y-H. Jia, Y. Mei, and M. Zhang, "A Bilevel Ant Colony Optimization Algorithm for Capacitated Electric Vehicle Routing Problem," *IEEE Transactions on Cybernetics*, vol. 52, no. 10, pp. 10855-10868, 2022, doi: 10.1109/TCYB.2021.3069942.

[22] A. Musdholifah and R. I. Kesuma, "Combination of ant colony optimization with local triangular kernel clustering for vehicle routing problem with time windows," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 16, no. 2, pp. 355-362, 2015, doi: 10.11591/ijeecs.v16.i2.pp355-362.

[23] Z. H. Ahmed, A. S. Hameed, M. L. Mutar, and H. Haron, "An Enhanced Ant Colony System Algorithm Based on Subpaths for Solving the Capacitated Vehicle Routing Problem," *Symmetry*, vol. 15, no. 11, 2023, doi: 10.3390/sym15112020.

[24] R. Hendrawan, Z. K. A. Baizal, and G. S. Wulandari, "Generating a multi-day travel itinerary recommendation using hybrid ant colony system and brainstorm optimization algorithm," *International Journal of Intelligent Engineering and Systems*, vol. 17, no. 2, pp. 223-234, 2024, doi: 10.22266/ijies2024.0430.20.

[25] S. Kadyrov, "CVRP with Genetic Algorithm," *GitHub*, Nov. 2024. [Online]. Available: https://github.com/shiralikadyrov/CVRP-with-Genetic-Algorithm. (Accessed: Nov. 26, 2024).

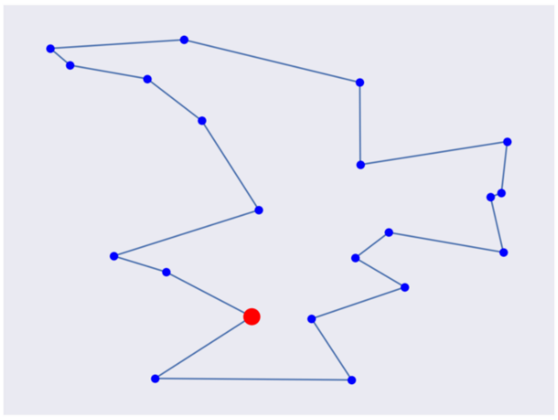
[26] H. Awad, R. Elshaer, A. AbdElmo'ez, and G. Nawara, "An effective genetic algorithm for capacitated vehicle routing problem," in *Proceedings of the International Conference on Industrial Engineering and Operations Management*, 2018, pp. 374-384.

[27] C.-H. Wang and J.-Z. Lu, "A hybrid genetic algorithm that optimizes capacitated vehicle routing problems," *Expert Systems with Applications*, vol. 36, no. 2, pp. 2921–2936, 2009, doi: 10.1016/j.eswa.2008.01.072

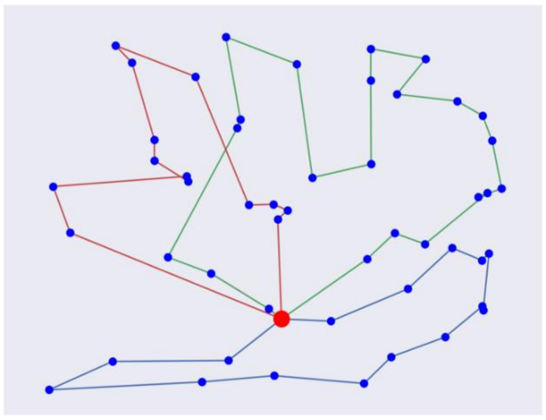
[28] C.-B. Cheng and K.-P. Wang, "Solving a vehicle routing problem with time windows by a decomposition technique and a genetic algorithm," Expert Systems with Applications, vol. 36, no. 4, pp. 7758–7763, 2009, doi: 10.1016/j.eswa.2008.09.001.

[29] Y. Shi, T. Boudouh, and O. Grunder, "A hybrid genetic algorithm for a home health care routing problem with time window and fuzzy demand," *Expert Systems with Applications*, vol. 72, pp. 160–176, 2017, doi: 10.1016/j.eswa.2016.12.013.

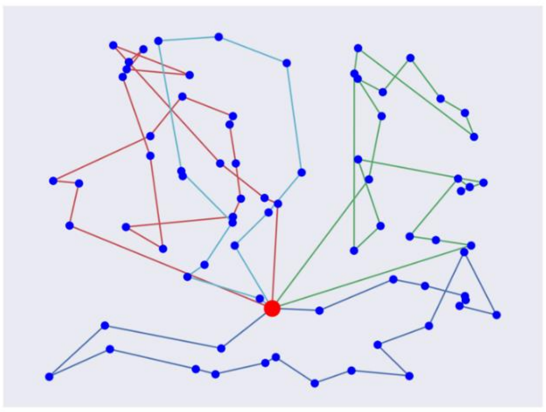
APPENDIX



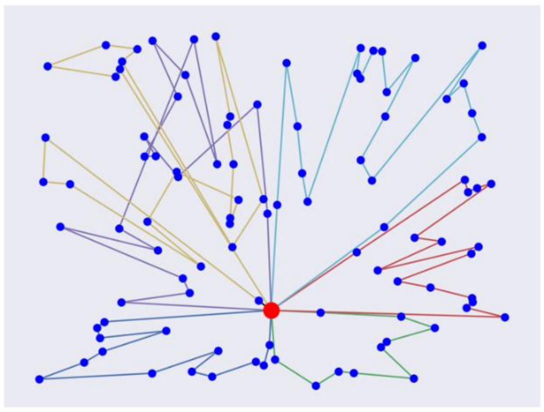
(a)



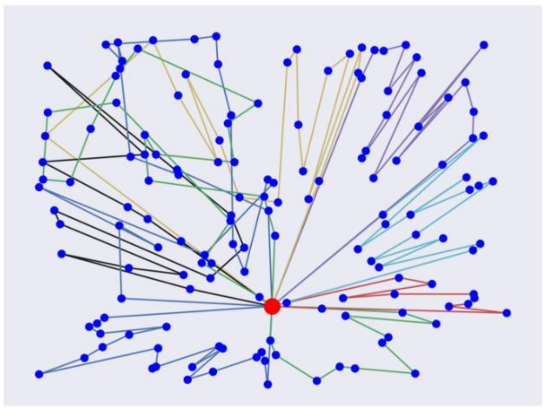
(b)



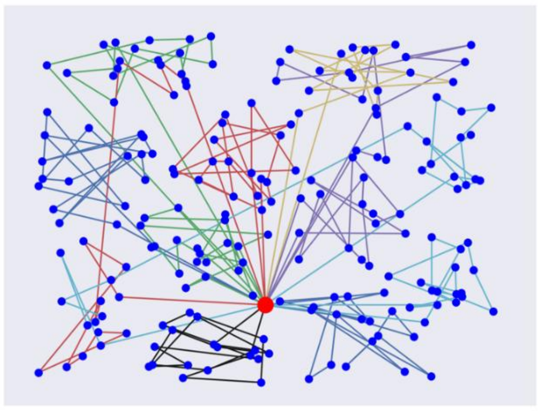
(c)



(d)



(e)

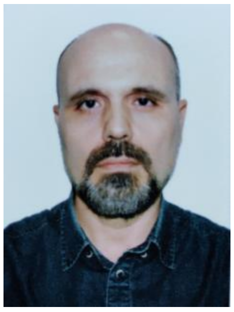


(f)

Figure 5. Heuristic route splitting based best solution for; (a) CVPR20 over 10k generations, (b) CVPR50 over 15k generations, (c) CVPR75 over 18k generations, (d) CVPR100 over 10k generations, (e) CVPR150 over 15k generations, and (f) CVPR200 over 11k generations

BIOGRAPHIES OF AUTHORS

Dr. Shirali Kadyrov is an associate professor at New Uzbekistan University, Tashkent, Uzbekistan. He served as a professor at Suleyman Demirel University, Kazakhstan, until 2024, having been a faculty member since 2017. From 2018 to 2019, he also held the position of Head of the Science Department. He earned his Bachelor's degree in Mathematics from Boğaziçi University, Turkey, in 2004 and his Ph.D. in Mathematics from The Ohio State University, USA, in 2010. His research interests focus on data science, dynamical systems, ergodic theory, and number theory. He can be contacted at email: sh.kadyrov@newuu.uz.



Dr. Cemil Turan is an associate professor at Suleyman Demirel University, Kazakhstan, where he has been a faculty member since 2008. From 2021-2022, he was also the Head of the Computer Science Department. He graduated with a bachelor's degree in Electrical Engineering from Yildiz Technical University, Turkey, in 1995, and completed his Ph.D. in Electrical and Computer Engineering from Mevlana Rumi University, Turkey, in 2016. His research interests are primarily in digital signal and image processing and computer vision. He can be contacted at email: cemil.turan@sdu.edu.kz.