JOURNAL OF INFORMATION AND COMMUNICATION TECHNOLOGY
https://e-journal.uum.edu.my/index.php/jict

How to cite this article:
Nabeel, N., Abdul-Rahman, S., & Wahid, J. (2025). An enhanced Red Deer algorithm with adaptive memory strategy for the capacitated vehicle routing problem. *Journal of Information and Communication Technology, 25*(2), 70-95. https://doi.org/10.32890/jict2026.25.2.4

An Enhanced Red Deer Algorithm with Adaptive Memory Strategy for the Capacitated Vehicle Routing Problem

[1] Noor Nabeel, [2]Syariza Abdul-Rahman & [3]Juliana Wahid
[1]Department of Information Management Techniques,
Technical College of Management, Northern Technical University, Iraq
[1&2]School of Quantitative Sciences, Universiti Utara Malaysia, Malaysia
[3]School of Computing, Universiti Utara Malaysia, Malaysia

[*1]noor.nabeel@ntu.edu.iq
[2]syariza@uum.edu.my
[3]w.juliana@uum.edu.my
*Corresponding author

Received: 5/5/2025 Revised: 26/6/2025 Accepted: 5/7/2025 Published: 30/4/2026

ABSTRACT

The vehicle routing problem (VRP), especially its capacitated (CVRP), has been extensively studied. In the CVRP, each customer must be served exactly once without exceeding the vehicle's capacity, with the aim of minimising the total distance of all routes. As a nondeterministic polynomial (NP)-hard problem, the CVRP is best solved using metaheuristics. The red deer algorithm (RDA), inspired by red deer mating behaviour, is population-based search metaheuristic with both exploration and exploitation phases. However, it may still get trapped in local optima and lacks a strong intensification mechanism for refining solutions. In RDA, not all solutions are explored, as it imposes no limit on solution reproduction, thereby restricting exploration. Thus, this study proposes an enhanced RDA that incorporates adaptive memory strategies to strengthen exploration and exploitation. The RDA classifies the population into males (i.e., high-quality solutions) and females or hinds (i.e., lower-quality solutions), with males further divided into commanders and stags. Exploitation occurs through roaring, selecting commanders, and fighting, while exploration involves forming harems and mating commanders with hinds both within and outside their harems. In enhanced RDA, previously mated hinds are adaptively excluded to promote mating with new ones, thereby enhancing exploration and solution quality. Enhanced RDA with an adaptive memory strategy (RDAM) was tested on 87 benchmark instances, outperforming a construction algorithm by 96.5% and the original RDA by 87.3%, and showing competitive results with established methods. The adaptive memory strategy in

RDA enhanced the total distance compared with the original RDA, while improving its exploration–exploitation in solving the CVRP, offering theoretical and practical benefits for logistics efficiency.

Keywords: Vehicle routing problem, metaheuristics, Red Deer algorithm, exploration and exploitation, memory structure.

INTRODUCTION

The vehicle routing problem (VRP) is about serving a set of customers that have different demands by a convoy of vehicles with the consideration of minimising the travelling cost (Dantzig & Ramser, 1959). There are several VRP variants like time windows, stochastic demand, stochastic travel and service times, and pickup and delivery. Among the various variants of the VRP, the fundamental form is the capacitated vehicle routing problem (CVRP), where customer demands must not exceed the capacity of the vehicles assigned to serve them. All vehicles have a certain capacity and start the journey from a single depot and return to it after serving the designated customers without passing through the same customer more than once. The aim of solving the CVRP is to serve customers with the shortest total distance while not violating any constraints. The CVRP is applicable to many cases, such as garbage collection (Mat et al., 2017; Zhang et al., 2020), tourist routing (Benjamin et al., 2019), and drug distribution (Mamoun et al., 2024).

CVRP is a nondeterministic polynomial (NP)-hard problem which can be solved by exact or approximate methods (Archetti et al., 2011). Exact methods are able to produce optimal solution, however the computational time is very expensive and limited to solving small problem dimensions (Toth & Vigo, 2002). Thus, researchers tend to employ metaheuristic approximate methods to solve CVRP with less computation time compared with exact algorithms (Dalbah et al., 2021). Metaheuristics have been found in wide-ranging applications, including electric power systems (Hamoodi et al., 2022; Al-Flaiyeh, 2024), cybersecurity (Alamiedy et al., 2020; Abdulhameed et al., 2024), Cancer Classification (Fajila & Yusof, 2025), waste collection (Sahib et al., 2025) (Khallaf et al., 2025), and nurse rostering (Ramli et al., 2020). Red deer algorithm (RDA) is a recent and promising metaheuristic algorithm inspired by the mating behaviour of red deer (Fathollahi-Fard et al., 2020). It has been successfully tested in problems such as the machine scheduling problem (MSP), travelling salesman (TSP), fixed charge transportation (FCT) and VRP (Fathollahi-Fard et al., 2020), demonstrating promising performance in exploring and identifying global solutions.

The RDA divides the population into males (representing high-quality solutions) and females or hinds (representing lower-quality solutions), with the male group further categorised into commanders and stags. One significant advantage of RDA is its adaptability, as its parameters can be fine-tuned to suit various problems and to effectively balance exploration and exploitation. Despite its strengths, one notable drawback of RDA is its tendency to allow repeated pairings during mating, which can lead to premature convergence and reduced solution diversity. As highlighted in Fathollahi-Fard et al. (2020), this limitation may hinder the algorithm's ability to fully explore the solution space, especially in complex or large-scale problems. However, the use of memory strategies in RDA remains limited, with only a study incorporating such mechanisms to enhance search efficiency and address challenges in solving large or complex problems (Zhou & Zong, 2021). The finding suggests that local search strategies and adaptive memory can significantly strengthen RDA, as noted in Fathollahi-Fard et al. (2020), yet its potential in other optimisation problems, such as the CVRP, remains underexplored.

This study proposes an enhanced RDA incorporating an adaptive memory strategy to strengthen its exploration and exploitation capabilities for effectively solving the CVRP. The adaptive memory strategy is integrated into several mating processes to regulate solution selection, ensuring more effective and diverse combinations. When a strong solution (i.e., commander) combines with some weaker ones (i.e., hinds) from its own group, each one it mates with is remembered to avoid repetition so soon. The same rule applies when it mates with hinds from other groups. If there are not enough new hinds left because some have already been used, the commander can reuse the ones in memory to reach the needed amount. This helps avoid repeating the same pairings adaptively and improves the overall search with diverse combinations. The enhanced RDA effectively solves the CVRP by balancing exploration and exploitation, producing robust, high-quality solutions that outperform existing methods on benchmark datasets. The results offer insights into RDA's effectiveness and potential in solving complex routing problems.

This study contributes to the field by extending the application of RDA to the CVRP, an area that has not been extensively explored in the literature. This study also advances the RDA by introducing an adaptive memory strategy during mating, preventing repeated pairings and promoting greater solution diversity. By enhancing both exploration and exploitation, the proposed RDA with an adaptive memory strategy addresses limitations of local search and premature convergence, enabling the algorithm to deliver more robust, higher-quality solutions for the CVRP. This paper begins with a review of the relevant literature, followed by a problem description and the mathematical formulation of the CVRP. It then introduces the proposed enhanced RDA with adaptive memory, presents and discusses the experimental results, and concludes with key findings and future research directions.

RELATED WORKS

RDA is a recent metaheuristic inspired by red deer mating behaviour introduced by (Fathollahi Fard et al., 2016). The algorithm is found to effectively explore promising regions of the search space and, in most cases, achieve global solutions by facilitating interaction between the intensification and diversification phases through parameter adjustment. It has shown strong performance across various optimisation and scheduling applications, consistently demonstrating its effectiveness in solving complex problems. The performance of RDA can be seen in comparative studies (Nasiria et al., 2017; Mohammadzadeh et al., 2018). The studies demonstrated that RDA outperforms several classical and modern optimisation algorithms in terms of solution quality and convergence rate across numerous test instances for the freight consolidation and containerization problem and truck scheduling problem in a cross-docking centre. Its applicability has further been validated in practical domains, such as the quay crane assignment problem (Fazli et al., 2019), complex function optimisation (Zitar & Abualigah, 2021), and the monoalphabetic cryptosystem (Jain et al., 2024). In addition, RDA has been successfully applied to various problems including VRP (Fathollahi-Fard et al., 2020).

Further enhancements to the RDA have been proposed in the literature, particularly through modified operators, hybrid approaches and adaptive mechanisms, to improve its effectiveness across various optimisation domains. For instance, the modified RDA (MRDA) proposed by Dey et al. (2021) and De et al. (2020) introduced adaptive adjustments by reducing parameters and using iteration data to enhance performance. It also modified the mating process, linking lower-quality solutions to good ones. These improvements enabled MRDA to outperform the original RDA, particle swarm optimisation (PSO), and genetic algorithm GA on benchmark image segmentation problems. Similarly, Nasiria et al. (2017) and Mohammadzadeh et al. (2018) applied the Taguchi method for parameter selection in RDA,

demonstrating its effectiveness in solving freight consolidation and containerisation problems, as well as truck scheduling in a cross-docking centre. Recent studies have explored hybrid forms of the RDA to improve its performance across different domains. For instance, the RDA was combined with simulated annealing (SA) in Dogani et al. (2020) to restore a water supply system, effectively handling unknown parameters in a multi-objective problem and outperforming a non-dominated sorting GA across various metrics.

Similarly, a hybrid red deer–salp swarm algorithm was developed for routing in wireless multimedia sensor networks by (Ambareesh & Madheswari, 2021), demonstrating more efficient path construction than competing methods. In the Internet of Nano-Things (IoNT), Gulec and Sahin, (2023) proposed an RDA-based distributed nano-sensor node clustering algorithm. More recently, Wang and Wu, (2024) integrated a multi-strategy improved RDA (MIRDA) with a bidirectional long short-term memory network (BiLSTM) for a fault warning model. MIRDA incorporates a population initialisation strategy, adaptive optimal guidance, a chaos-regulation factor, and a double-sided mirror-reflection theory. Studies on the incorporation of memory within the RDA remain limited. A work by Zhou and Zong (2021) is the only study that applied an adaptive memory-based RDA, focusing on a truck scheduling problem with cross-docking and time windows. Their approach addressed RDA's weaknesses in local search and computation time by integrating oscillating local search with scatter search and path relinking, leading to improved intensification and better convergence on large instances. The objective was to minimise total product transhipment time, along with the earliness and tardiness costs for outbound trucks. However, the application of adaptive memory in other optimisation problems, such as the CVRP, remains scarce. This highlights the potential for further exploration of adaptive memory integration within RDA to enhance its performance in solving complex routing problems.

Adaptive memory is designed to strengthen metaheuristic search processes by storing, updating, and reusing information from previously visited solutions. Unlike memoryless algorithms, which may repeatedly explore the same regions of the search space, adaptive memory provides a structured mechanism to guide the search, enhancing both intensification and diversification. Originating from tabu search (Glover, 1989), where memory structures prevent revisiting recently explored solutions while encouraging new exploration, adaptive memory has since been widely adopted in frameworks such as scatter search (Glover, 1977) and path relinking (Glover, 1998) to systematically combine high-quality solutions and avoid premature convergence. Although the original RDA can produce optimal solutions, research on its extensions remains limited, offering opportunities for further exploration and development, particularly in the context of CVRP benchmark instances.

Despite its strengths, RDA has notable shortcomings in local search strategies and the use of adaptive memory, as highlighted in (Fathollahi-Fard et al., 2020). In the original RDA, repeated pairings during mating led to premature convergence and reduced solution diversity. This limitation restricts the algorithm's ability to fully explore the solution space, especially in complex or large-scale problems. The list of previous studies on RDA and its enhancement is presented in Table 1.

To evaluate the performance of the proposed RDA-based approach in a comparative experiment, several advanced metaheuristic algorithms are used as high-validation benchmarks. Among those, the ant colony optimization (ACO) by Gao et al. (2023) for solving the CVRP. The main breakthrough of this proposed algorithm is its optimisation using ACO's self-organization and positive feedback to address ACO's inherent slow initial convergence and tendency towards local optima. The ACO is built by an elite ant strategy for superior local search, the max-min ant system for controlling the pheromone, and a dual pheromone update mechanism. The algorithm has shown competitive performance against state-

of-the-art. Thammano and Rungwachira (2021) combined a modified ant system (MAS), a sweep algorithm, path relinking, and local search operators to enhance CVRP solution quality. Initially, the sweep algorithm creates capacity-feasible routes by grouping customers based on their polar angle and adjusting sub-routes for feasibility. MAS then generates new solutions using a max–min pheromone update with three elitist options (global best, iteration best, second-best) and a bias parameter to ensure diversified search. Path relinking intensifies the search by exploring trajectories between elite solutions using swaps, insertions, reversals, and greedy local searches. The approach was tested on 72 standard CVRP instances, this hybrid approach performs competitively against state-of-the-art algorithms, even improving the best-known solution for one instance, with deviations typically under 1.71%.

Table 1

Studies Related to RDA and Its Enhancements

Author	Problem	Modification to RDA
Nasiria et al. (2017)	Freight consolidation and containerization problem	Parameters selection using Taguchi
Mohammadzadeh et al. (2018)	Truck scheduling problem in a cross-docking centre	Parameters selection using Taguchi
Fazli et al. (2019)	Coordinated quay crane scheduling and assignment problem	No modification
Fathollahi-Fard et al. (2020)	Single machine scheduling, traveling salesman, fixed charge transportation and VRP	No modification
De et al. (2020)	Multi-level Image Thresholding	Adaptive strategy on operators and parameters
Dogani et al. (2020)	Water supply system	Hybrid RDA-SA
Dey et al. (2021)	Multilevel image segmentation	Adaptive strategy on operators and parameters
Abu Zitar (2021)	Review paper	No modification
Zhou and Zong (2021)	Cross-dock truck scheduling with products time window	Heuristic oscillating local search and adaptive memory
Zitar and Abualigah (2021)	Optimizing complex functions	No modification
Ambareesh and Madheswari (2021)	Wireless multimedia sensor networks	Hybrid red deer salp swarm
Gulec and Sahin (2023)	IoNT	RDA based distributed nano-sensor node clustering
Jain et al. (2024)	Monoalphabetic cryptosystem	No modification
Wang and Wu (2024)	Fault warning model	MIRDA with a BiLSTM network

Aiming to highlight the strengths and limitations of the RDA with adaptive memory employed in this study, its performance was evaluated against the following methods. GA is used by Sajid et al. (2020) as the global search component in a hybrid genetic and simulated annealing (HGSA) framework where chromosomes are permutation-based giant tours, tournament selection is applied, swap mutation maintains diversity, and a novel nearest-neighbour crossover operator is introduced to favour connecting customers that are geographically close so as to reduce long edges in the tour. The GA with the nearest-neighbour crossover is also evaluated independently, and across 86 CVRP benchmark instances the solutions are consistently improved further by HGSA, highlighting both the strength of the tailored GA representation and operators and the benefit of hybridising GA with local search to better explore neighbourhood solutions.

GA was adopted by Sbai et al. (2022) to address the CVRP by integrating GA with variable neighbourhood search (VNS). The VNS procedure was embedded in the GA's mutation operator to balance global exploration and local exploitation. The proposed hybrid GA–VNS was evaluated on well-known benchmark instances and real-world data from the Tunisian post office, demonstrating competitive solution quality compared to methods from previous studies. The results confirmed that hybridisation hybridisation of metaheuristics is an effective strategy for solving large-scale CVRP instances.

Altabeeb et al. (2021) proposed a cooperative hybrid firefly algorithm (CHFA) for the CVRP. CHFA combines the firefly algorithm with local search (refined 2-opt), genetic operators (three permutation-based mutations including 2-h-opt and partially matched crossover) to achieve an optimal mix of intensification and diversity. Multiple hybrid firefly populations run simultaneously within a multi-population "island" system, periodically exchanging solutions to encourage successful search patterns while maintaining diversity. When tested on 102 CVRP benchmark instances, CHFA has consistently achieved near-optimal or new best-known solutions, outperforming previous versions of the firefly algorithm and many cutting-edge metaheuristics in both solution quality and convergence speed. Perturbation-based variable neighbourhood search with adaptive selection mechanism (PVNSASM) by Faiz et al. (2018) is a VNS-based metaheuristic for enhancing intensification–diversification balance for the CVRP. The customers are removed by several ruin operators (random, relatedness, long-arc-broken) and then reinserted via multiple greedy and regret-based insertion heuristics, while ASM updates weights of ten perturbation schemes based on their empirical success in finding new best solutions and selects them via a roulette-wheel rule. The method was applied to the A and B benchmark sets. PVNSASM outperformed a basic VNS in both average deviation from best-known solutions and CPU time, and it is competitive with other advanced CVRP heuristics.

Kır et al. (2017) proposed a combined array of tabu search and adaptive large neighbourhood search for solving the CVRP. This method introduced three major parts: the specific destroy/repair operators that take both the route sequence and vehicle capacity into account, a search operator which clusters service points, and a cluster-based diversification strategy to escape local optima. Computational experiments based on standard A, M, and G benchmark sets indicate that the algorithm can find all known optima for small instances, achieve gaps of about 0.01% for larger ones, and generate new best solutions for three very large ones. Ng et al. (2017) implemented Multiple Colonies Artificial Bee Colony (MC-ABC) which allowed random inter-colony information transmission during the scout phase, thus striking a better trade-off between exploration and exploitation compared to single-colony ABC variants. Computational experiments on classical CVRP benchmarks (Augerat A, B, M, P sets) show that MC-ABC consistently outperforms the original and modified ABC in deviation from best-known solutions, especially for larger instances.

THE PROPOSED METHOD

Problem Description

In CVRP, each vehicle serves a subgroup of customers' demands that less or equal vehicle capacity and each customer should be served only once by a single vehicle. CVRP composes I customers $C=\{c_1, c_2, c_3, \ldots, c_I\}$ with a specific demand for each one $R=\{r_1, r_2, r_3, \ldots, r_I\}$ subject to J vehicles $V=\{v_1, v_2, v_3, \ldots, v_J\}$ identical in capacity $Q=\{q_1, q_2, q_3, \ldots, q_J\}$. The customers and depot located according to their coordinates $X=\{x_0, x_1, x_2, \ldots, x_I\}$ and $Y=\{y_0, y_1, y_2, \ldots, y_I\}$ since x_0, y_0 represent depot coordinates. The distance between each customer, including the depot, and other customers is calculated using Euclidean distance as in Equation 1:

$$d_{c_i,c_k} = \sqrt{(x_k - x_i)^2 + (y_k - y_i)^2} \tag{1}$$

The distances are assumed symmetric. The sequence of route in each vehicle is $s=\{c_0, c_1, c_2, \ldots, c_0\}$ begins and ends at c_0 the depot. In addition, the load of vehicle is verified to not exceed its capacity as in Equation 2. Equations 3 and 4 are to validate visiting each customer only once. Equation 5 is to ensure each vehicle starts and ends at the depot.

$$\sum_{k=1}^{n} \quad \leq r_k \leq q_j, n \in I \tag{2}$$

$$\sum_{k=1}^{J} \quad \sum_{e=0}^{I} \quad X_{ep}^{k} = 1, p = 1,2, \ldots, I \tag{3}$$

$$\sum_{k=1}^{J} \quad \sum_{p=0}^{I} \quad X_{ep}^{k} = 1, e = 1,2, \ldots, I \tag{4}$$

$$\sum_{p=1}^{I} \quad X_{0p}^{k} = \sum_{p=1}^{I} \quad X_{p0}^{k} = 1, k \in J \tag{5}$$

The cost of route s_j served by vehicle v_j is calculated by Equation 6:

$$ej=d_{0,1}+dn,0+\sum_{k=1}^{n-1} \quad d_{k,k+1}, n \in I \tag{6}$$

To find the total cost, which is the objective function that needs to be minimised, the costs of all vehicles are summed together as in Equation 7:

$$Te = \sum_{j=1}^{J} \quad e_j \tag{7}$$

The RDA

The RDA is a population-based metaheuristic that iteratively improves solution quality through search and optimisation. Its main steps are described in Algorithm 1. The main steps of the proposed methodology are discussed in more detail. These steps outline the overall workflow of RDA and clarify how each step contributes to solving the problem.

Population Partitioning

In this study, the initial population for the RDA was generated using a two-phase constructive algorithm that minimises total travel distance. In the first phase, customers were sorted in descending order and sequentially assigned to vehicles, ensuring capacity constraints were not violated. If some customers remained unassigned due to limited space, neighbourhood structures were applied to the existing routes to create space for them, resulting in a feasible base solution in which all customers were served within

vehicle capacity. In the second phase, additional neighbourhood structures were used to diversify the solutions and form the initial population. The population was then divided into two groups based on solution quality: males, representing the top-performing solutions, and hinds (i.e., females), representing the remaining ones.

Subsequently, several operations were performed on the male group (roaring and fighting), followed by mating operations involving both males and hinds. In this study, the population consisted of 15% males and 85% hinds to ensure that each superior male had sufficient hinds for multiple pairings during the mating phases.

Algorithm 1. RDA

Input: Dataset instance
Analyse instance data
Initialization: Generate initial population

Population Partitioning: The population is divided into males (better solutions) and females (weaker solutions)

Iterations: For i=1 to max iterations
Roaring Phase: Males generate neighbour solutions to improve fitness and are then divided into commanders (top γ%) and stags (remaining males)

Fighting Phase: Commanders compete with stags, producing new solutions; the best replace the weaker ones

Mating with Own Harem: Commanders mate with a random α% of their own hinds to generate better offspring.

Mating with Other Harems: Commanders also mate with β% of hinds from other harems to increase diversity

Mating by Stags: Non-commander males' mate with hinds of similar quality to maintain population diversity

End iteration

Output: Best solution found

Roaring Phase

In nature, male red deer roar to expand their territory, though the attempt may succeed or fail. Similarly, in the RDA, after classifying individuals as males and females, male solutions undergo the roaring phase, in which two neighbourhood structures, move and swap, are applied. A newly generated solution replaces the previous one if it is feasible and yields a better objective value. The choice of neighbourhood structure is determined by a random number between 0 and 1: values between 0 and 0.33 trigger a move, between 0.33 and 0.66 a swap, and between 0.66 and 1 a sequential move.

The move and sequential move operations differ in how they select source and destination nodes. In the move operation, both are chosen randomly, excluding the depot (node 0) to avoid infeasible solutions. The resulting neighbour solution must also satisfy vehicle capacity constraints, as illustrated in Figure 1. In contrast, the sequential move systematically selects source nodes (excluding depots) and experiments with possible destinations in sequence, ensuring that each move reduces total cost without violating capacity limits. For the swap operation, two non-depot customers are randomly selected and exchanged between routes. The new solution must remain feasible for both vehicles involved, as shown in Figure 2. After the roaring phase, the top γ% of males are designated as commanders, while the remaining males are classified as stags ($males - (\gamma \times males)$).

Figure 1

Move Neighbourhood Structure

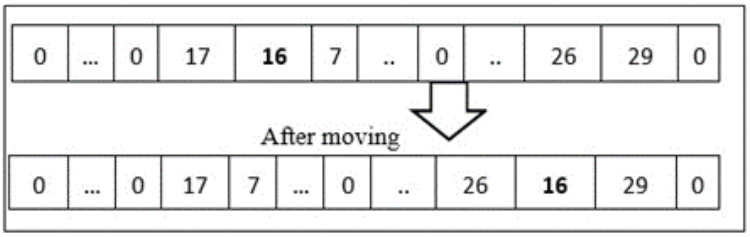
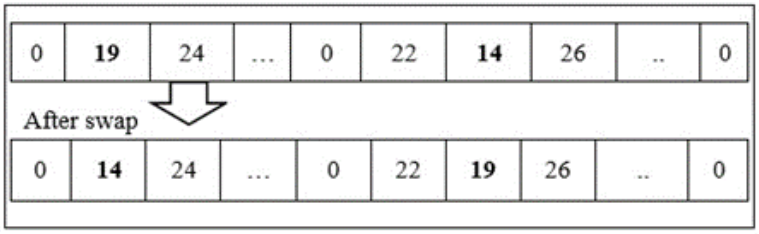


Figure 2

Swap Neighbourhood Structure



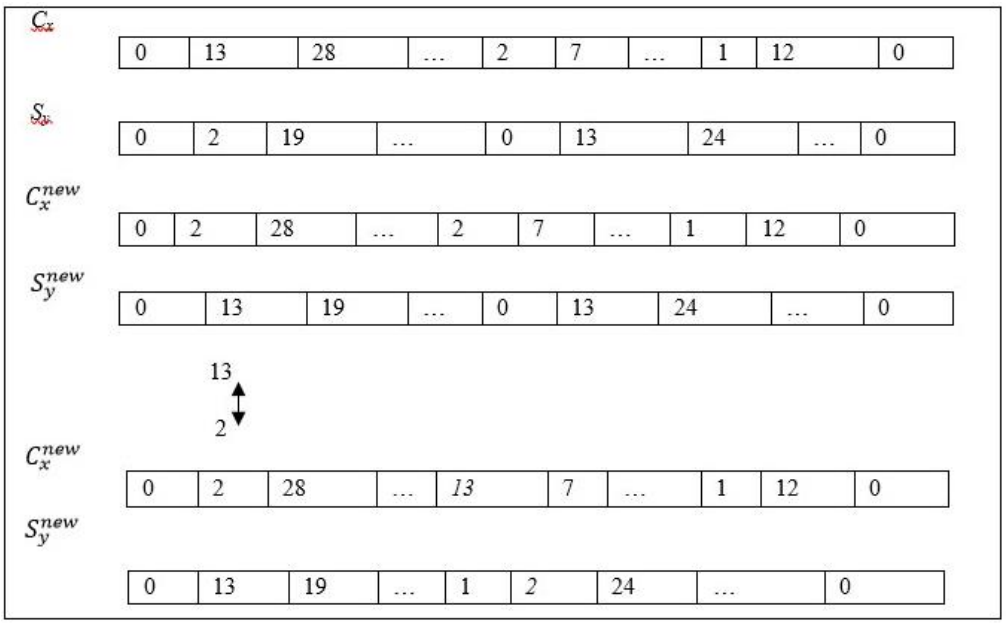
Fighting Phase

After identifying commanders and stags in the roaring phase, each commander engages in a fight with a randomly selected stag. A cross-swap neighbourhood structure is applied to both, generating two new solutions. If the new commander's solution yields a better objective value than the original and the new stag's solution remains feasible under CVRP constraints, both replace their previous counterparts. In some cases, even if the new stag's solution is inferior, it will still replace the old stag to maintain diversity.

As shown in Figure 3, a commander C_x and a stag S_y are randomly selected to exchange customers. For example, customer (13) from C_x is swapped with customer (2) from S_y, producing two new solutions, C_x^{new} and S_y^{new}. To satisfy the constraint that each customer is visited only once, customer (2) in C_x^{new} is replaced by (13), and customer (13) in S_y^{new} is replaced by (2). The cross-swap is considered successful if C_x^{new} is feasible and yields a better objective value than C_x, while S_y^{new} remains feasible.

Figure 3

Cross-Swap Neighbourhood Structure



Harem Formation

Each commander forms a harem by seizing a certain number of hinds, with stronger commanders obtaining more hinds. The number of hinds assigned to each commander is determined by calculating the commander's priority (cp_n) based on its solution quality (cq_n), as shown in Equation 8, where *max(c_q)* represents the worst solution in the population:

$$cp_n = cq_n - max(c_q) \tag{8}$$

The normalized power of each commander ($cpow_n$) is then computed to represent its relative strength among all *A* commanders, as shown in Equation 9:

$$cpow_n = \left| \frac{cp_n}{\sum_{i=1}^{A} cp_i} \right| \tag{9}$$

Finally, the number of hinds assigned to each commander's harem is determined using Equation 10, which increases proportionally with the commander's power:

$$commander_n^{hinds} = round(\ cpow_n * RDhinds) \tag{10}$$

At the end of this process, each commander possesses its own harem consisting of a group of hinds.

Mating with Own Harem

During the mating phase, each commander mates with α percent of the hinds in his harem, as defined in Equation 11:

$$commander_n^{mate}=round(\alpha*commander_n^{hinds}) \quad (11)$$

where $commander_n^{mate}$ represents the number of hinds that commander n will mate with. The hinds are selected randomly, and a cross-swap neighbourhood structure is applied between each commander and the selected hind. If the resulting offspring solution is feasible and achieves a better objective value than the commander, the offspring replaces the commander as the new solution.

Mating with Other Harems

Commanders mate with β percent of hinds with another randomly selected commander's harem $command_k^{hinds}$ as in Equation 12:

$$command_n^{mate\_k}=round(\beta*command_k^{hinds}) \quad (12)$$

In Equation 12 $command_n^{mate\_k}$ is the number of hinds from $commander_k$ harem that $commander_n$ is going to mate with. Cross-swap neighbourhood structure is applied on the mating between commanders and hinds in this step.

Mating by Stags

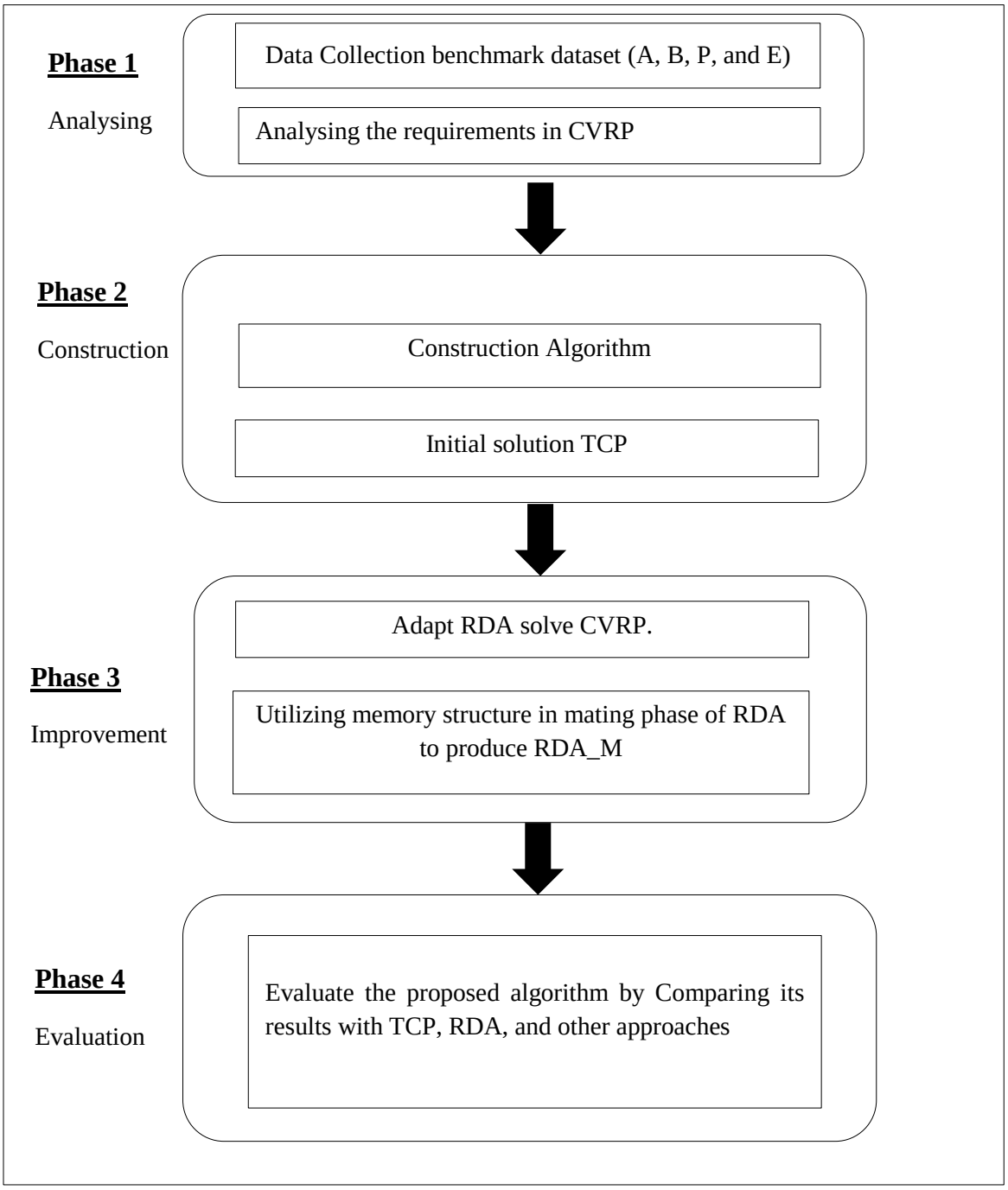
Each stag can mate with the nearest ***hind*** regardless of which harem the hind belongs to. In this study, the nearest hind to a specific stag is defined as the one whose objective function value has the smallest difference compared to that of the stag, among all available hinds. Cross-swap neighbourhood structure is applied in this step, and if the new solution is feasible and has a better objective function value than the stag's, then it will replace the stag while keeping a feasible hind regardless of the hind's new objective function value.

RDA with Memory

The original RDA (Fathollahi Fard et al., 2016a) lacks a memory mechanism to manage mating and ensure wider coverage of hinds. In this study, memory is introduced to enhance exploration by storing previously visited solutions and excluding them from future mating, thus directing the search toward unvisited solutions. The memory is adaptive, with its size changing as commanders mate with new hinds. It is applied in both mating stages. Each hind involved in mating is stored in memory to prevent repeated pairing. If the number of available hinds becomes less than β% due to exclusions, the commander may mate with hinds from the memory to maintain diversity. The methodology of the study is illustrated in Figure 4 and the pseudocode of the RDA with memory is presented in Algorithm 2.

Figure 4

Methodology of the study



Algorithm 2. RDA with Memory

Parameter settings: population_size, number of males (*m*), number of hinds (*h*), Maximum Improvisation (MI), γ, α, and β
Population initialization
Evaluate initial solutions to classify them by quality
While MI not met do
  For (*i*=0 to *m*)
// Roar the male and replace it if a better solution is found
    r =random (0,1); if(r>=0 and r<=0.33) swap; if(r>0.33 and r<=0.66) move; if(r>0.66 and r<=1.0) seq_move;}
  End for;
// Reorder males to identify commanders and stags
// Fight between commanders and stags
  For (*i*=0 to *number_of_commanders)* cross_swap(random(commander), random(stag));
    Accept better solutions and reorder males
  End for
// Commanders form harems
  For (j=0 to number_of_commanders)
    Calculate (cp_j, cq_j, $cpow_j$, $commander_j^{hinds}$);
  End for
//Mate
  For (j=0 to number_of_commanders) mated_hinds_counter=0; vector alpha_mem;
    While commander j mated_hinds_counter <= round(α* $commander_j^{hinds}$)
      Select hind *h* randomly from commander *j* harem
      If (h not in alpha_mem)
        Cross_swap($commander_j$, *h*);
        Add h to alpha_mem; mated_hinds_counter++
    End while;
    Clear alpha_mem;
  End for
  For (j=0 to number_of_commanders)
      mated_hinds_counter=0; vector beta_mem;
Select another harem randomly k
While commander j mated_hinds_counter <= round(β * $commander_k^{hinds}$)
// Select hind h randomly from commander k harem
      If (h not in beta_mem or (h in beta_mem and no unmated hinds in j))
        Cross_swap($commander_j$, *h*);
        if (h NOT exist in beta_mem)
          Add h to beta_mem;
          mated_hinds_counter++;
        End if;
  End for
  For *n*=0 to number_of_stags
    Find the $hind_{nearest}$ with objective value closest to the $stag_n$
    Cross_swap($stag_n$, $hind_{nearest}$)
  End for
End while
Return best solution

Parameters Setting of RDA

The RDA requires five parameters: population size, number of males and females, commander percentage (γ), and two mating ratios: α for a commander's own harem and β for other harems. The parameter values follow Fathollahi-Fard et al. (2020) and are listed in Table 2.

Table 2

RDA Parameters

Parameter	Value
Population size	100
The number of males	15
The number of females	85
γ	0.7
The number of stags	4
α	0.9
β	0.4

EXPERIMENTAL RESULTS AND ANALYSIS

The original RDA (Fathollahi Fard et al., 2016) and the enhanced RDA with adaptive memory structure (RDAM) were implemented in C++ and executed on a computer equipped with a Core i5-6300U 2.4 GHz processor. Experiments for both algorithms were conducted using CVRP benchmark datasets A, B, P, and E, with ten independent runs for each instance. The results were compared with those of the Construction Algorithm (CA) and other improvement algorithms reported in the literature (Gao et al., 2023; Thammano & Rungwachira, 2021; Sajid et al., 2020; Sbai et al., 2022; Altabeeb et al., 2021; Ng et al., 2017). These datasets (Capacitated Vehicle Routing Problem Library, 2025), include problem instances with 13 to 101 customers and between 4 and 14 vehicles. Tables 3 to 6 and Figures 5 to 8 show the minimum solution cost for RDA, RDAM, and CA on datasets A, B, P, and E respectively. A lower solution cost among the algorithms indicates a better solution. In the tables, the best results of RDAM, compared with the original RDA and CA, are in bold, while bold italics highlight the better result when comparing the original RDA and CA. The underlined values indicate the instances where optimal solutions were achieved.

Table cells marked with (–) indicate that no value is available for the corresponding instance. In addition, t-scores at α=0.05 are presented for comparisons between RDA and CA, and between RDAM and RDA. As shown in Table 3, the original RDA outperformed CA in 26 out of 27 instances in dataset A (except A-n39-k6), while RDAM surpassed CA in all 27 instances. Compared to RDA, RDAM performed better in 25 instances, equal in one, and worse in one (A-n45-k6). RDAM achieved the best in 25 instances. The t-test results confirm that RDA significantly outperforms CA across all instances, while RDAM achieves statistically significant improvements over RDA in 20 instances. Figure 5 further demonstrates RDAM's superiority over both CA and RDA in most cases. Table 4 demonstrates the superior performance of RDA and RDAM over CA across 23 instances in dataset B. RDAM outperformed RDA in 21 instances, showing notable improvement. RDAM was the best in 21 cases. The t-test results indicate that RDA achieved significant gains over CA in all instances, while RDAM was significantly better than RDA in 21 instances. Figure 6 further highlights RDAM's strong performance across most instances, with RDA ranking second.

Table 5 presents the results for dataset P, where RDA outperformed CA in 19 of 24 instances and matched CA in 2, while RDAM surpassed CA in 22 of 24 instances and matched CA in 2. Compared with RDA, RDAM performed better in 20 instances and was equal in 3, achieving the best results in 19 cases. RDA achieved 22 significant improvements over CA, while RDAM recorded 17 significant improvements over RDA. Additionally, RDAM produced two optimal solutions (P-n19-k2 and P-n22-k8), whereas RDA obtained one (P-n22-k8). Figure 7 illustrates the superior performance of RDAM compared to both CA and RDA. Table 6 presents the performance of the methods on dataset E. RDA outperformed CA in 12 out of 13 instances, while RDAM was superior to CA in 12 instances and equal in one. Compared to RDA, RDAM performed better in 10 cases and was equal in 3. RDAM was the best in 9 and achieved 3 optimal solutions, while RDA achieved 2 optimal solutions. RDA showed significant improvements in 12 instances, whereas RDAM achieved 10 significant improvements over RDA. Both RDAM and CA reached the optimal solution for E-n13-k4.

As illustrated in Figure 8, RDAM consistently produced high-quality solutions across most instances compared with CA and RDA. From the results in Tables 3 to 6, RDAM achieved the greatest improvement over RDA in dataset A (25/27; 92.5%), followed by dataset B (21/23; 91.3%), dataset P (20/24; 83.3%), and dataset E (10/13; 76.9%). The percentage of improved instances for each dataset is calculated using Equation 13.

$$\frac{Significant\ instances}{Total\ instances} \times 100 \tag{13}$$

It is noteworthy that both RDA and RDAM achieved the exact two optimal solutions in dataset E; hence, RDAM could not further improve upon RDA in those cases. The order of improvement percentages aligns with the level of dataset complexity, represented by the ratio of total demands to $(q \times K)$, as described by Uchoa et al. (2017). The average complexity values for datasets A, B, P, and E are 92.25, 91.65, 93.75, and 90.92, respectively. Therefore, datasets with lower complexity exhibited greater improvements by RDAM. The percentage of significant solutions among all improved instances is computed using Equation 14.

$$\frac{Significant\ instances}{Improved\ instances} \times 100 \tag{14}$$

The percentage of significantly improved solutions by RDAM over RDA is 80% (20/25), 100% (21/21), 85% (17/20), and 100% (10/10) for datasets A, B, P, and E, respectively. These findings show that most improvements achieved by RDAM are statistically significant compared to RDA. Both RDA and RDAM demonstrated significant reductions in solution cost compared to CA across all datasets, except for instance E-n13-k4. Overall, RDAM achieved four optimal solutions and one better-than-optimal solution, outperforming RDA in most instances, while RDA obtained three optimal solutions.

Table 3

Performance of RDA, RDAM, and CA of Dataset A

Instance	RDA	RDAM	CA	t-score (RDA, CA)	t-score (RDAM, RDA)	Opt
A-n32-k5	***799***	**787**	840	6.19	2.35	784
A-n33-k5	***662***	662	683	6.08	-1.86	661
A-n33-k6	***746***	**743**	758	5.34	2.78	742

(continued)

Instance	RDA	RDAM	CA	t-score (RDA, CA)	t-score (RDAM, RDA)	Opt
A-n34-k5	***791***	**780**	810	4.46	-2.45	778
A-n36-k5	***809***	**802**	863	6.52	1.81	799
A-n37-k5	***682***	**672**	705	4.13	2.46	669
A-n37-k6	***959***	**951**	974	5.11	1.54	949
A-n38-k5	***745***	**734**	819	6.27	0.35	730
A-n39-k5	***850***	**842**	868	6.00	2.64	822
A-n39-k6	849	**840**	842	4.85	0.73	831
A-n44-k6	***969***	**946**	1024	6.00	4.08	937
A-n45-k6	***1014***	1022	1183	9.72	0.34	944
A-n45-k7	***1162***	**1156**	1186	7.57	0.20	1146
A-n46-k7	***938***	**925**	990	6.04	1.74	914
A-n48-k7	***1114***	**1074**	1217	10.13	3.02	1073
A-n53-k7	***1065***	**1036**	1252	18.31	3.73	1010
A-n54-k7	***1216***	**1187**	1354	11.10	3.88	1167
A-n55-k9	***1099***	**1082**	1305	12.21	3.06	1073
A-n60-k9	***1412***	**1375**	1475	6.64	5.47	1354
A-n61-k9	***1118***	**1087**	1281	7.97	3.79	1034
A-n62-k8	***1345***	**1320**	1469	7.97	4.87	1288
A-n63-k9	***1684***	**1651**	1832	7.80	7.15	1616
A-n63-k10	***1368***	**1333**	1446	5.10	6.84	1314
A-n64-k9	***1495***	**1442**	1540	6.263	6.51	1401
A-n65-k9	***1251***	**1198**	1455	8.64	5.98	1174
A-n69-k9	***1241***	**1194**	1302	5.17	7.40	1159
A-n80-k10	***1868***	**1821**	1991	4.06	7.21	1763

Figure 5

Performance Analysis on Dataset A

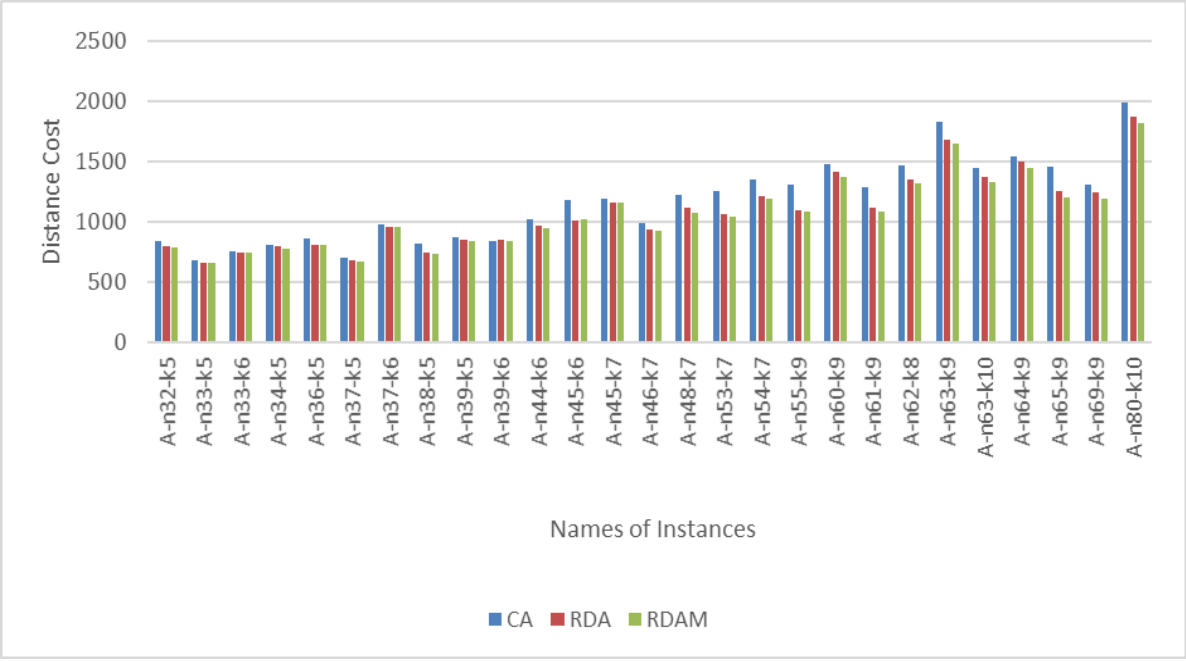


Table 4

Performance of RDA, RDAM, and CA of Dataset B

Instance	RDA	RDAM	CA	t-score (RDA, CA)	t-score (RDAM, RDA)	Opt
B-n31-k5	***678***	**676**	682	4.83	8.42	672
B-n34-k5	***803***	**790**	814	5.57	7.21	788
B-n35-k5	***959***	**956**	1002	4.61	5.23	955
B-n38-k6	***812***	**809**	835	4.81	5.45	805
B-n39-k5	***565***	**553**	596	6.01	6.90	549
B-n41-k6	***866***	**834**	996	9.28	8.29	829
B-n43-k6	***766***	**749**	793	6.44	5.86	742
B-n44-k7	***924***	929	995	12.61	7.78	909
B-n45-k5	***780***	**757**	861	8.44	4.70	751
B-n45-k6	***755***	**694**	843	12.24	1.33	678
B-n50-k7	***774***	**744**	806	7.34	5.70	741
B-n50-k8	***1328***	**1317**	1388	8.23	4.81	1312
B-n51-k7	***1112***	**1048**	1359	11.21	1.94	1032
B-n52-k7	***771***	**754**	897	11.14	3.80	747
B-n56-k7	***732***	**718**	838	11.56	5.72	707
B-n57-k7	***1398***	**1401**	1623	12.39	-1.50	1153
B-n57-k9	***1641***	**1611**	1749	9.75	6.17	1598
B-n63-k10	***1600***	**1527**	1699	10.35	5.43	1496
B-n64-k9	***941***	**892**	1200	12.22	5.30	861
B-n66-k9	***1372***	**1334**	1651	12.95	5.09	1316
B-n67-k10	***1099***	**1068**	1279	9.59	6.86	1032
B-n68-k9	***1325***	**1301**	1358	5.84	5.80	1272
B-n78-k10	***1321***	**1263**	1428	8.98	6.56	1221

Figure 6

Performance Analysis on Dataset B

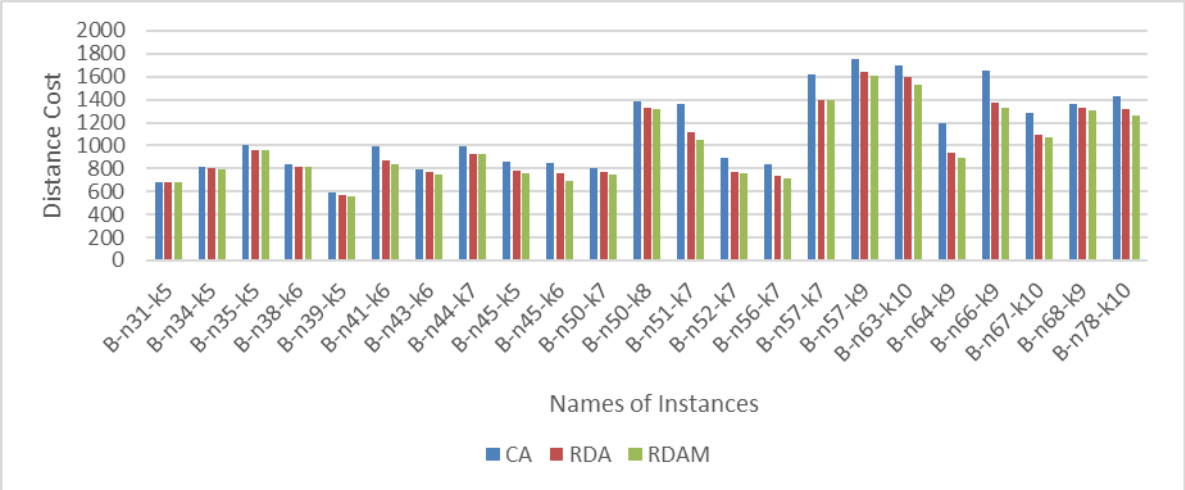


Table 5

Performance of RDA, RDAM, and CA of Dataset P

Instance	RDA	RDAM	CA	t-score (RDA, CA)	t-score (RDAM, RDA)	Opt
P-n16-k8	***454***	451	451	0.47	3.67	450
P-n19-k2	***219***	**212**	237	4.34	1.24	212
P-n20-k2	***218***	218	249	4.82	0.41	216
P-n21-k2	212	212	212	2.79	3.35	211
P-n22-k2	***217***	217	218	4.58	0.48	216
P-n22-k8	603	**600**	603	2.74	3.08	603
P-n23-k8	533	**531**	533	3.45	-0.62	529
P-n40-k5	***465***	**461**	469	4.17	5.51	458
P-n45-k5	***527***	**512**	545	4.72	6.72	510
P-n50-k7	***572***	**568**	598	4.71	1.87	554
P-n50-k8	***687***	696	801	7.63	1.41	631
P-n50-k10	***715***	**707**	765	5.94	1.50	696
P-n51-k10	783	**753**	865	7.44	2.76	741
P-n55-k7	***595***	**584**	608	4.17	10.45	568
P-n55-k8	***608***	**607**	655	5.42	5.72	588
P-n55-k10	790	**709**	766	0.37	25.32	694
P-n55-k15	***1066***	**1042**	1168	8.18	0.28	989
P-n60-k10	***770***	**768**	888	10.23	3.74	744
P-n60-k15	***996***	**971**	1085	7.42	5.45	968
P-n65-k10	***823***	**808**	881	7.12	3.20	792
P-n70-k10	***897***	**861**	1055	12.61	10.17	827
P-n76-k4	***657***	**629**	745	7.27	5.60	593
P-n76-k5	***693***	**653**	779	9.98	2.54	627
P-n101-k4	***773***	**720**	796	4.45	3.98	681

Figure 7

Performance Analysis on Dataset P

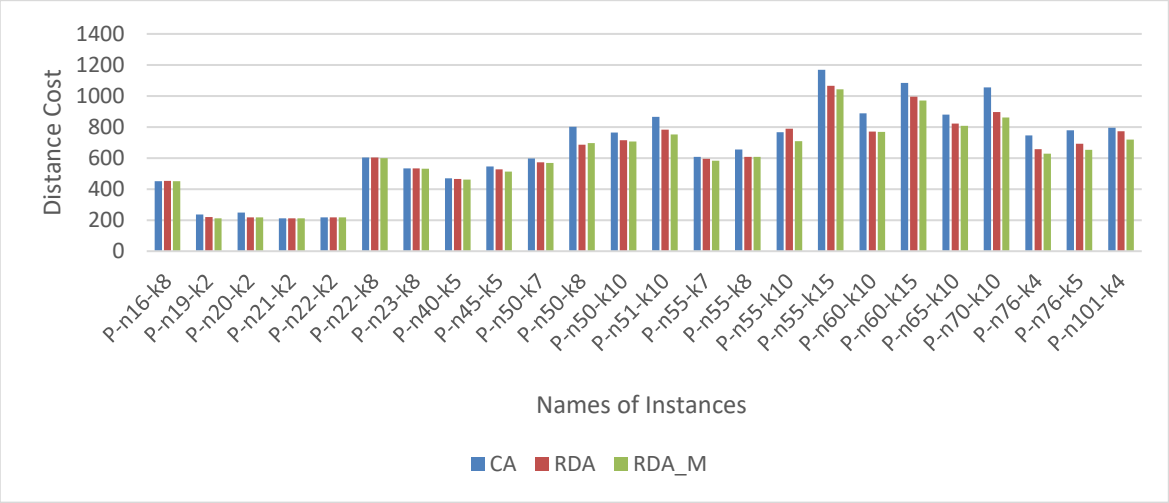


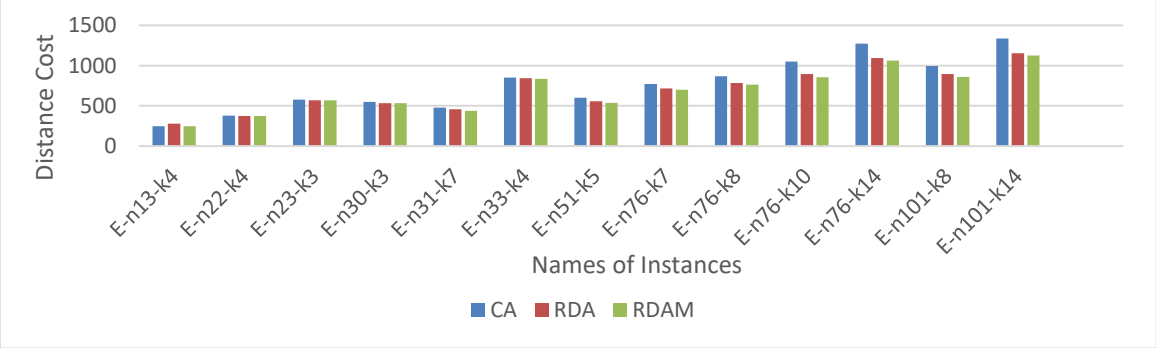
Table 6

Performance of RDA, RDAM, and CA of Dataset E

Instance	RDA	RDAM	CA	t-score (RDA, CA)	t-score (RDAM, RDA)	Opt
E-n13-k4	277	247	247	-1.99	155.08	247
E-n22-k4	***375***	375	379	19.04	2.42	375
E-n23-k3	***569***	569	577	8.38	1.83	569
E-n30-k3	***535***	535	550	34.38	1.61	534
E-n31-k7	***456***	**438**	477	6.38	-0.66	379
E-n33-k4	***843***	**837**	850	12.77	3.51	835
E-n51-k5	***558***	**537**	602	17.17	6.12	521
E-n76-k7	***717***	**701**	770	56.57	1.74	682
E-n76-k8	***782***	**764**	869	53.53	1.37	735
E-n76-k10	***894***	**857**	1051	54.06	6.00	830
E-n76-k14	***1096***	**1064**	1273	74.66	4.52	1021
E-n101-k8	***894***	**860**	994	64.85	10.12	817
E-n101-k14	***1152***	**1126**	1338	80.64	3.21	1071

Figure 8

Performance Analysis on Dataset E



Comparison of RDAM with Metaheuristics Methods

The RDAM is compared with other population-based methods from the literature that have been applied in solving the CVRP, using either complete or partial data sets. The value of RDAM is highlighted in bold if the value is better than that of other methods in at least one instance. The best result is highlighted by bold and italic while if an instance was not addressed by the compared methods, it is indicated with (–) and (Opt) is the optimal value. In Tables 7, 8, 9, and 10, RDAM is compared with the ACO algorithm (Gao et al., 2023), MAS (Thammano & Rungwachira, 2021), GA1 (Sajid et al., 2020), GA2 (Sbai et al., 2022), CHFA (Altabeeb et al., 2021), PVNSASM (Faiz et al., 2018), TALNS (Kır et al., 2017), and MC-ABC (Ng et al., 2017). As shown in Table 7, RDAM outperformed ACO and GA1 across all addressed instances in dataset A, achieving 13, 1, 3 (out of 14), and 4 better solutions than MAS, CHFA, MC-ABC, and GA2, respectively. RDAM also produced results comparable to PVNSASM and TALNS. For dataset B (Table 8), RDAM achieved 22, 10, 21, 1, 10 (out of 22), and 9 better solutions then ACO, MAS, GA1, CHFA, MC-ABC, and GA2, respectively. However, RDAM did not outperform PVNSASM on any of the 17 instances tested.

In Table 9, for dataset P, RDAM produced better solutions in 18 (from 22), 10 (out of 22), 18 (from 23), 2 (from 23), 11 (from 23), and 8 (from 23) instances compared with ACO, MAS, GA1, CHFA, MC-ABC, and GA2, respectively. As shown in Table 10, RDAM outperformed GA1 across all 11 instances in dataset E and produced two (out of 10) better solutions than GA2. However, RDAM did not surpass CHFA on the eight instances considered. These findings indicate that RDAM is generally competitive across datasets P and E, though its performance is less dominant against CHFA in specific cases. Moreover, RDAM achieved one optimal solution and one better-than-optimal solution (P-n22-k8) in dataset P, as well as three optimal solutions in dataset E. In comparison, MC-ABC also produced a better-than-optimal solution for P-n22-k8 in dataset P but required one additional vehicle. For instance, P-n55-k8, MC-ABC outperformed the known optimal by reducing the number of routes used.

Table 7

Comparison of RDAM with Other Metaheuristics on Dataset A

Instance	RDAM	ACO	MAS	GA1	FA	ABC	GA2	PVNS ASM	TALNS	Opt
A-n32-k5	**787**	865	813	830	796	***784***	***784***	787	***784***	784
A-n33-k5	**662**	738	668	674	***661***	***661***	***661***	***661***	***661***	661
A-n33-k6	**743**	832	743	767	***742***	748	744	***742***	***742***	742
A-n34-k5	**780**	877	779	807	***778***	-	784	***778***	***778***	778
A-n36-k5	**802**	921	818	859	***799***	-	***799***	***799***	***799***	799
A-n37-k5	**672**	778	693	755	***669***	***669***	673	***669***	***669***	669
A-n37-k6	**951**	1079	953	998	***949***	963	***949***	951	***949***	949
A-n38-k5	**734**	786	738	754	***730***	-	***730***	***730***	***730***	730
A-n39-k5	**842**	960	832	858	***822***	-	834	***822***	***822***	822
A-n39-k6	**840**	984	834	889	***831***	***831***	***831***	833	***831***	831
A-n44-k6	**946**	1097	***937***	-	***937***	-	***937***	942	939	937
A-n45-k6	**1022**	1034	***948***	-	953	952	***948***	***948***	955	944
A-n45-k7	**1156**	1332	1164	1202	1147	***1146***	1152	***1146***	1153	1146
A-n46-k7	**925**	1319	934	1005	***914***	935	927	***914***	915	914
A-n48-k7	**1074**	1306	1123	1162	***1073***	***1073***	***1073***	***1073***	***1073***	1073
A-n53-k7	**1036**	1172	1047	1088	1011	-	***1010***	-	1026	1010
A-n54-k7	**1187**	1337	1169	1263	1172	-	***1167***	-	1169	1167
A-n55-k9	**1082**	-	1076	1130	1074	***1073***	1079	1074	1074	1073
A-n60-k9	**1375**	1582	1399	1464	1355	1370	***1354***	1360	1366	1354
A-n61-k9	**1087**	1231	1051	1146	1039	-	***1034***	1042	1045	1034
A-n62-k8	**1320**	1489	1338	1407	1298	-	***1290***	-	1302	1288
A-n63-k9	**1651**	2102	1623	1745	1630	-	***1619***	1629	1644	1616
A-n63-k10	**1333**	1601	1327	1435	***1314***	-	***1314***	-	1325	1314
A-n64-k9	**1442**	1636	1453	1551	1420	-	***1407***	-	1442	1401
A-n65-k9	**1198**	1440	1178	1246	1178	***1174***	1180	1180	1189	1174
A-n69-k9	**1194**	1388	1175	1287	***1162***	-	1164	-	1169	1159
A-n80-k10	**1821**	2205	1801	1864	1773	1820	***1768***	1784	1790	1763

Table 8

Comparison of RDAM with Other Metaheuristics on Dataset B

Instance	RDAM	ACO	MAS	GA1	FA	ABC	GA2	PVNSASM	Opt
B-n31-k5	**676**	725	***672***	681	***672***	***672***	691	***672***	672
B-n34-k5	**790**	864	789	814	***788***	***788***	***798***	***788***	788
B-n35-k5	**956**	1105	963	988	***955***	***955***	979	-	955
B-n38-k6	**809**	886	806	838	806	807	828	***805***	805
B-n39-k5	**553**	655	560	566	550	550	573	***549***	549
B-n41-k6	**834**	870	***829***	850	***829***	836	847	830	829
B-n43-k6	**749**	779	749	776	***742***	746	762	***742***	742
B-n44-k7	**929**	1018	922	965	***909***	924	***909***	***909***	909
B-n45-k5	**757**	841	755	788	***751***	***751***	***751***	***751***	751
B-n45-k6	**694**	740	691	734	686	699	***678***	680	678
B-n50-k7	**744**	859	746	806	***741***	742	***741***	***741***	741
B-n50-k8	**1317**	1417	1331	1375	1318	1328	1321	***1313***	1312
B-n51-k7	**1048**	1092	***1032***	1042	***1032***	***1032***	1042	-	1032
B-n52-k7	**754**	842	771	788	***747***	752	758	-	747
B-n56-k7	**718**	893	727	755	709	718	718	***707***	707
B-n57-k7	**1401**	1388	1164	1193	-	***1153***	***1153***	-	1153
B-n57-k9	**1611**	1777	1651	1667	1610	1637	***1598***	-	1598
B-n63-k10	**1527**	1734	1557	1616	1503	1569	***1496***	-	1496
B-n64-k9	**892**	997	868	946	***862***	896	868	-	861
B-n66-k9	**1334**	1475	1337	1391	***1319***	1355	1321	1320	1316
B-n67-k10	**1068**	1217	1075	1123	1042	1077	***1039***	1042	1032
B-n68-k9	**1301**	1436	1298	1342	***1278***	1315	1292	1287	1272
B-n78-k10	**1263**	1446	1245	1318	***1224***	1299	1247	1242	1221

Table 9

Comparison of RDAM with Other Metaheuristics on Dataset P

Instance	RDAM	ACO	MAS	GA1	FA	ABC	GA2	Opt
P-n16-k8	451	***450***	***450***	451	***450***	***450***	***450***	450
P-n19-k2	***212***	215	***212***	223	***212***	***212***	***212***	212
P-n20-k2	**218**	218	***216***	221	***216***	***216***	***216***	216
P-n21-k2	**212**	225	***211***	224	***211***	***211***	***211***	211
P-n22-k2	**217**	228	***216***	230	***216***	***216***	***216***	216
P-n22-k8	**600**	630	603	***597***	***590a***	603	603	603
P-n23-k8	**531**	631	529	542	***529***	***529***	***529***	529
P-n40-k5	**461**	544	467	498	***458***	***458***	***458***	458
P-n45-k5	**512**	610	529	559	***510***	516	***510***	510
P-n50-k7	**568**	684	587	625	***554***	565	575	554
P-n50-k8	**696**	690	658	684	***631***	633	657	631
P-n50-k10	**707**	798	705	769	***697***	**713**	722	696
P-n51-k10	**753**	870	746	799	***742***	768	773	741
P-n55-k7	**584**	704	601	632	***568***	587	***568***	568
P-n55-k8	**607**	796	588	-	***576b***	588	-	588
P-n55-k10	**709**	-	705	761	***698***	708	***698***	694

(continued)

Instance	RDAM	ACO	MAS	GA1	FA	ABC	GA2	Opt
P-n55-k15	1042	-	-	**1014**	-	-	***989***	989
P-n60-k10	**768**	891	772	829	***749***	773	757	744
P-n60-k15	**971**	1107	990	1061	***968***	1002	988	968
P-n65-k10	**808**	986	827	887	***792***	828	810	792
P-n70-k10	**861**	1022	844	948	***827***	875	851	827
P-n76-k4	**629**	798	625	687	***593***	625	622	593
P-n76-k5	**653**	833	643	724	***628***	662	647	627
P-n101-k4	**720**	1001	-	819	***681***	721	717	-

Table 10

Comparison of RDAM with Other Metaheuristics on Dataset E

Instance	RDAM	GA1	FA	GA2	Opt
E-n13-k4	***247***	-	-	-	247
E-n22-k4	***375***	393	***375***	379	375
E-n23-k3	***569***	**580**	***569***	***569***	569
E-n30-k3	**535**	547	***534***	545	534
E-n31-k7	438	-	-	-	379
E-n33-k4	**837**	864	***835***	***835***	835
E-n51-k5	**537**	599	***521***	***521***	521
E-n76-k7	**701**	821	***683***	693	682
E-n76-k8	**764**	853	***738***	750	735
E-n76-k10	**857**	951	-	***830***	830
E-n76-k14	1064	1156	-	***1039***	1021
E-n101-k8	**860**	1009	-	***825***	817
E-n101-k14	**1126**	1264	***1082***	-	1071

Table 11 illustrates the role of the adaptive memory structure in regulating commanders' mating behaviour during the mating phase of the proposed RDA. The analysis is based on a partial result involving 11 commanders mating with β% of hinds from other harems, tested on instance A-n32-k5. These 11 commanders were selected from a population of 15 males, with 70% designated as commanders. The "Mated Hinds" column shows the hinds each commander mated with, while the "Prevention Times" column reflects how often the adaptive memory successfully prevented repeated mating with previously selected hinds during this phase. Commander 2 mated with no hinds due to randomly selecting a harem of a weak commander that has no hinds or approximately zero β%. Among the commanders, 7 experienced at least one instance of mating prevention, indicating that the memory mechanism was actively engaged in avoiding repeated selection of the same hinds. Notably, Commander 11 recorded the highest number of prevention events (5), suggesting a strong overlap in selected hinds, likely due to a concentration in high-quality candidates. Commanders 3, 4, and 6 also had multiple prevention events (2–3 times), while others, like Commanders 1 and 10, experienced only a single prevention. In contrast, Commanders 2, 7, 8, and 9 showed no prevention events, possibly due to more varied or less competitive selection. These findings highlight the effectiveness of adaptive memory in promoting mating diversity, reducing overexploitation, and encouraging broader exploration of the solution space by blocking repeated pairings.

Table 11

Mating Prevention Times for Commanders in the Mating Phase

Commander ID	Mated hinds	Prevention times
1	20, 36, 24, 93	1
2	-	-
3	59, 85, 89, 58, 57, 66	2
4	47, 58, 57, 61	2
5	69, 27	1
6	63, 42, 21, 91, 70	3
7	94	0
8	17, 30	0
9	87	0
10	31, 53	1
11	51, 45, 16, 96, 40, 75, 50	5

CONCLUSION

In this study, an enhanced Red Deer Algorithm with adaptive memory (RDAM) was developed to effectively solve the CVRP by explicitly embedding CVRP constraints into every search operator and guiding the population toward high-quality, feasible routes. RDAM starts with a two-phase constructive procedure that assigns customers to vehicles while respecting capacity, then uses neighbourhood moves to create a fully feasible and diversified initial population in which all customers are served within vehicle capacity. During the roaring and fighting phases, move, sequential move, swap, and cross-swap neighbourhoods are repeatedly applied to male solutions, but new routes are only accepted if they remain feasible and reduce the total travel cost, thereby steadily improving CVRP solutions while preserving CVRP constraints. The adaptive memory mechanism is then activated during the mating phases: each hind that mates with a commander is stored in memory and temporarily excluded from further pairings, forcing commanders to mate with new hinds from their own and other harems, thereby exploring new routing patterns while still enforcing feasibility through cross-swap. Through this balance of intensification and diversification, RDAM significantly reduced total route distance on 87 CVRPLIB instances, outperforming the construction algorithm by 96.5% and the original RDA by 87.3%, and achieving four optimal and one better-than-optimal solution while remaining competitive with several established metaheuristics. Despite these encouraging results, several limitations should be acknowledged. First, adaptive memory is applied only during the mating phase, while the fighting and stag phases remain memoryless. Second, RDAM needs to integrate with local search methods to further strengthen exploitation and robustness. Finally, the experimental evaluation is restricted to classical symmetric CVRP instances with up to 101 customers and identical vehicles, so scalability to larger real-world problems and richer variants (e.g., time windows, heterogeneous fleet, multi-depot) remains untested. Addressing these limitations by extending memory to additional phases, embedding stronger local search, and scaling to larger and richer CVRP variants offers promising directions for future research and further improvement.

ACKNOWLEDGMENT

The authors would like to express their gratitude to the Ministry of Higher Education (MoHE) Malaysia for supporting this study under the Fundamental Research Grant Scheme (FRGS/1/2021/ICT02/ UUM/02/5), as well as RIMC, Universiti Utara Malaysia, for the study's administration.

REFERENCES

Abdulhameed, A. A., Alazawi, S. A. H., & Hassan, G. M. (2024). An optimized model for network intrusion detection in the network operating system environment. *Mesopotamian Journal of CyberSecurity*, *4*(3), 75–85. https://doi.org/10.58496/MJCS/2024/017

Alamiedy, T. A., Anbar, M., Alqattan, Z. N. M., & Alzubi, Q. M. (2020). Anomaly-based intrusion detection system using multi-objective grey wolf optimisation algorithm. *Journal of Ambient Intelligence and Humanized Computing*, *11*(9), 3735–3756. https://doi.org/10.1007/s12652-019-01569-8

Al-Flaiyeh, M. (2024). Influence of using ACO algorithm in STATCOM performance. *NTU Journal of Engineering and Technology*, *3*(1). https://doi.org/10.56286/ntujet.v3i1.821

Altabeeb, A. M., Mohsen, A. M., Abualigah, L., & Ghallab, A. (2021). Solving capacitated vehicle routing problem using cooperative firefly algorithm. *Applied Soft Computing*, *108*. https://doi.org/10.1016/j.asoc.2021.107403

Ambareesh, S., & Madheswari, A. N. (2021). HRDSS-WMSN: A multi-objective function for optimal routing protocol in wireless multimedia sensor networks using hybrid red deer salp swarm algorithm. *Wireless Personal Communications*, *119*(1), 117–146. https://doi.org/10.1007/s11277-021-08201-z

Archetti, C., Feillet, D., Gendreau, M., & Grazia Speranza, M. (2011). Complexity of the VRP and SDVRP. *Transportation Research Part C: Emerging Technologies*, *19*(5), 741–750. https://doi.org/10.1016/j.trc.2009.12.006

Benjamin, A. M., Abdullah, A. S., Abdul-Rahman, S., Nazri, E. M., & Yahaya, H. Z. (2019). Developing a comprehensive tour package using an improved greedy algorithm with tourist preferences. *Journal of Sustainability Science and Management*, *14*(4), 106–117. https://jssm.umt.edu.my/wp-content/uploads/sites/51/2020/05/9.14.4pdf.pdf

Dalbah, L. M., Al-Betar, M. A., Awadallah, M. A., & Zitar, R. A. (2021). A modified coronavirus herd immunity optimizer for capacitated vehicle routing problem. *Journal of King Saud University - Computer and Information Sciences*. https://doi.org/10.1016/j.jksuci.2021.06.013

Dantzig, G. B., & Ramser, J. H. (1959). The truck dispatching problem. *Management Science*, *6*(1), 80–91. https://doi.org/10.1287/mnsc.6.1.80

De, S., Dey, S., Debnath, S., & Deb, A. (2020). *A new modified red deer algorithm for multi-level image thresholding.* 2020 Fifth International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN), 105–111. https://doi.org/10.1109/ICRCICN50933.2020.9296166

Dey, S., De, S., Deb, A., & Debnath, S. (2021). *Multilevel image segmentation using modified red deer algorithm.* 2021 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence), 8, 362–368. https://doi.org/10.1109/Confluence51648.2021.9377112

Dogani, A., Dourandish, A., Ghorbani, M., & Shahbazbegian, M. R. (2020). A hybrid meta-heuristic for a bi-objective stochastic optimization of urban water supply system. *IEEE Access*, *8*, 135829–135843. https://doi.org/10.1109/ACCESS.2020.3009885

E. Nasiria, A. J. & Afsharia, M. H.-K. (2017). Addressing the freight consolidation and containerization problem by recent and hybridized meta-heuristic algorithms. *International Journal of Engineering*, *30*(3). https://doi.org/10.5829/idosi.ije.2017.30.03c.10

Faiz, A., Subiyanto, & Arief, U. M. (2018). An efficient meta-heuristic algorithm for solving capacitated vehicle routing problem. *International Journal of Advances in Intelligent Informatics*, *4*(3), 212–225. https://doi.org/10.26555/ijain.v4i3.244

Fajila, F., & Yusof, Y. (2025). Mutable composite firefly algorithm for microarray-based cancer classification. *Journal of Information and Communication Technology*, *24*(1), 102–129. https://doi.org/10.32890/jict2025.24.1.5

Fathollahi Fard, A. M., Hajiaghaei-Keshteli, M., & Tavakkoli-Moghaddam, R. (2016). *Red Deer Algorithm (RDA); A new optimization algorithm inspired by red deers' mating.* 12Th International Conference on Industrial Engineering, January, 1–10.

Fathollahi-Fard, A. M., Hajiaghaei-Keshteli, M., & Tavakkoli-Moghaddam, R. (2020). Red deer algorithm (RDA): A new nature-inspired meta-heuristic. *Soft Computing*, *24*(19), 14637–14665. https://doi.org/10.1007/s00500-020-04812-z

Fazli, M., Fathollahi-Fard, A. M., & Tianc, G. (2019). Addressing a coordinated quay crane scheduling and assignment problem by red deer algorithm. *International Journal of Engineering, Transactions B: Applications*, *32*(8), 1186–1191. https://doi.org/10.5829/ije. 2019.32.08b.15

Gao, Y., Wu, H., & Wang, W. (2023). A hybrid ant colony optimization with fireworks algorithm to solve capacitated vehicle routing problem. *Applied Intelligence*, *53*(6), 7326–7342. https://doi.org/10.1007/s10489-022-03912-7

Glover, F. (1977). Heuristics for integer programming using surrogate constraints. *Decision Sciences*, *8*(1), 156–166. https://doi.org/10.1111/j.1540-5915.1977.tb01074.x

Glover, F. (1989). Tabu Search—Part I. *ORSA Journal on Computing*, *1*(3), 190–206. https://doi.org/10.1287/ijoc.1.3.190

Glover, F. (1998). *A template for scatter search and path relinking* (pp. 1–51). https://doi.org/10.1007/BFb0026589

Gulec, O., & Sahin, E. (2023). Red Deer Algorithm based nano-sensor node clustering for IoNT. *Journal of Network and Computer Applications*, *213*, 103591. https://doi.org/10.1016/j.jnca.2023.103591

Hamoodi, A. N., Abdulla, F. S., & Ahmed Alwan, A. (2022). Maximizing output power for solar panel using grey wolf optimization. *NTU Journal of Engineering and Technology*, *1*(4). https://doi.org/10.56286/ntujet.v1i4.164

Jain, A., Bansal, S., Das, N. N., & Gupta, S. S. (2024). Red deer algorithm to detect the secret key of the monoalphabetic cryptosystem. *Soft Computing*, *28*(17–18), 10569–10582. https://doi.org/10.1007/s00500-024-09849-y

Khallaf, N., Abdel-Raouf, O., Hadhoud, M., Dawam, M., & Kafafy, A. (2025). A deep reinforcement learning and fractional packing framework for routing and scheduling in healthcare waste supply chains. *Supply Chain Analytics*, *12*, 100164. https://doi.org/10.1016/j.sca.2025.100164

Kır, S., Yazgan, H. R., & Tüncel, E. (2017). A novel heuristic algorithm for capacitated vehicle routing problem. *Journal of Industrial Engineering International*, *13*(3), 323–330. https://doi.org/10.1007/s40092-017-0187-9

Mamoun, K. A., Hammadi, L., Ballouti, A. El, Novaes, A. G. N., & Cursi, E. S. De. (2024). Vehicle routing optimization algorithms for pharmaceutical supply chain: A systematic comparison. *Transport and Telecommunication*, *25*(2), 161–173. https://doi.org/10.2478/ttj-2024-0012

Mat, N. A., Benjamin, A. M., Abdul-Rahman, S., & Wibowo, A. (2017). *Nearest greedy for solving the waste collection vehicle routing problem: A case study*. 040018. https://doi.org/10.1063/1.5012206

Mohammadzadeh, H., Sahebjamnia, N., Fathollahi-Fard, A. M., & Hahiaghaei-Keshteli, M. (2018). New approaches in metaheuristics to solve the truck scheduling problem in a cross-docking center. *International Journal of Engineering, Transactions B: Applications*, *31*(8), 1258–1266. https://doi.org/10.5829/ije.2018.31.08b.14

Ng, K. K. H., Lee, C. K. M., Zhang, S. Z., Wu, K., & Ho, W. (2017). A multiple colonies artificial bee colony algorithm for a capacitated vehicle routing problem and re-routing strategies under time-dependent traffic congestion. *Computers & Industrial Engineering*, *109*, 151–168. https://doi.org/10.1016/j.cie.2017.05.004

Ramli, R., Ahmad, S. N. I., Abdul-Rahman, S., & Wibowo, A. (2020). A tabu search approach with embedded nurse preferences for solving nurse rostering problem. *International Journal for Simulation and Multidisciplinary Design Optimization*, *11*, 10. https://doi.org/10.1051/smdo/2020002

Sahib, T. M., Mohd-Mokhtar, R., & Mohd-Kassim, A. (2025). Ant colony optimization algorithm with sequential variable neighbourhood search change step in the waste collection system. *Jurnal Teknologi (Sciences & Engineering)*, *87*(4), 651–662. https://doi.org/10.11113/jurnalteknologi.v87.19726

Sajid, M., Jafar, A., & Sharma, S. (2020). *Hybrid genetic and simulated annealing Algorithm for capacitated vehicle routing problem.* In PDGC 2020 - 2020 6th International Conference on Parallel, Distributed and Grid Computing (pp. 131–136). https://doi.org/10.1109/PDGC50313.2020.9315798

Sbai, I., Krichen, S., & Limam, O. (2022). Two meta-heuristics for solving the capacitated vehicle routing problem: The case of the Tunisian Post Office. *Operational Research*, *22*(1), 507–549. https://doi.org/10.1007/s12351-019-00543-8

Thammano, A., & Rungwachira, P. (2021). Hybrid modified ant system with sweep algorithm and path relinking for the capacitated vehicle routing problem. *Heliyon*, *7*(9). https://doi.org/10.1016/j.heliyon.2021.e08029

Toth, P., & Vigo, D. (2002). 8. VRP with Backhauls. In P. Toth & D. Vigo (Eds.), *The Vehicle Routing Problem* (Vols. 2007-Janua, pp. 195–224). Society for Industrial and Applied Mathematics. https://doi.org/10.1137/1.9780898718515.ch8

Wang, Y., & Wu, M. (2024). Efficient fault warning model using improved red deer algorithm and attention-enhanced bidirectional long short-term memory network. *Processes*, *12*(10), 2253. https://doi.org/10.3390/pr12102253

Zhang, S., Mu, D., & Wang, C. (2020). A solution for the full-load collection vehicle routing problem with multiple trips and demands: An application in Beijing. *IEEE Access*, *8*, 89381–89394. https://doi.org/10.1109/ACCESS.2020.2993316

Zhou, B., & Zong, S. (2021). Adaptive memory red deer algorithm for cross-dock truck scheduling with products time window. *Engineering Computations*, *38*(8), 3254–3289. https://doi.org/10.1108/EC-05-2020-0273

Zitar, R. A., & Abualigah, L. (2021). *Application of red deer algorithm in optimizing complex functions.* 2021 14th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI), 1–6. https://doi.org/10.1109/CISP-BMEI53629. 2021.9624345