A Comparative Analysis of Metaheuristic Algorithms for the Vehicle Routing Problem with Time Windows in E-commerce Last-Mile Delivery

FIRST A. AUTHOR[1], SECOND B. AUTHOR[1], AND THIRD C. AUTHOR[1]

[1]*Department of Industrial Engineering and Management, R.V. College of Engineering, Bengaluru 560059, Karnataka, India*

Corresponding author: First A. Author (e-mail: author@rvce.edu.in).

This work was supported by the Department of Industrial Engineering and Management, R.V. College of Engineering, under the Experiential Learning component of the Optimization Techniques course (IM266TER), supervised by Dr. Vivekanand S. Gogi.

ABSTRACT The rapid expansion of electronic commerce has intensified the complexity of last-mile delivery, which now accounts for nearly 40 to 53 percent of total logistics expenditure in urban supply chains. The Vehicle Routing Problem with Time Windows (VRPTW) is a well-established NP-hard combinatorial optimization problem that lies at the core of efficient last-mile operations, yet the relative performance of competing metaheuristic algorithms under unified experimental conditions remains inadequately benchmarked in the modern e-commerce context. This study presents a comparative analysis of three widely adopted metaheuristic algorithms, namely the Genetic Algorithm (GA), Ant Colony Optimization (ACO), and Simulated Annealing (SA), alongside Google OR-Tools as an industry-grade baseline solver. All four approaches were evaluated on the Solomon benchmark dataset covering clustered, random, and random-clustered instances of 100 customers each. Performance was measured across four dimensions: total travel distance, number of vehicles deployed, computational runtime, and convergence behaviour. Across thirty independent experimental runs per instance, Google OR-Tools consistently delivered the shortest computation times and best solutions on clustered configurations, while the Genetic Algorithm produced the most competitive solution quality across mixed configurations. Ant Colony Optimization exhibited strong performance on structured problems with distinct cluster patterns, and Simulated Annealing provided a lightweight alternative suited to resource-constrained environments. Statistical significance of pairwise differences was confirmed using the Wilcoxon signed-rank test with a 95 percent confidence threshold. The findings offer actionable guidance for e-commerce logistics managers regarding algorithm selection based on problem characteristics, computational budget, and scalability requirements, and highlight emerging hybrid machine-learning and metaheuristic approaches as a promising direction for future dynamic-routing research.

INDEX TERMS Ant Colony Optimization, benchmarking, combinatorial optimization, e-commerce logistics, Genetic Algorithm, Google OR-Tools, last-mile delivery, metaheuristics, NP-hard problems, optimization techniques, routing algorithms, Simulated Annealing, Solomon benchmark, supply chain management, vehicle routing problem.

I. INTRODUCTION

The rapid expansion of electronic commerce across global and emerging markets has fundamentally transformed consumer expectations regarding delivery speed, reliability, and cost. In India alone, the e-commerce logistics market is projected to surpass USD 160 billion by 2028, driven by increased internet penetration, the rise of quick-commerce platforms, and a growing preference for doorstep delivery. The number of daily parcel deliveries across major Indian metropolitan areas has crossed ten million, with a compound annual growth rate exceeding 22 percent over the past five years. Within this rapidly evolving ecosystem, last-mile delivery represents the final and most complex segment of the supply chain, accounting for approximately 40 to 53 percent of total logistics cost due to low drop density, traffic unpredictability, and tight customer service windows.

The economic impact of last-mile inefficiency is substantial. A mid-sized e-commerce hub dispatching 500 parcels per day across a fleet of 20 vehicles can incur losses of approximately 12 to 18 lakh Indian rupees per year due to suboptimal routing alone, manifested as excess fuel consumption, missed delivery windows, driver overtime, and customer churn. As fleet sizes scale into the hundreds and customer counts exceed several thousand per depot, the magnitude of these losses grows nonlinearly, making algorithmic optimization a strategic imperative rather than a tactical convenience.

The Vehicle Routing Problem (VRP), first formalized by Dantzig and Ramser in 1959, remains the theoretical foundation for last-mile delivery optimization. In its canonical form, the VRP seeks to determine the minimum-cost set of routes for a fleet of vehicles originating from a central depot to serve a set of geographically dispersed customers. The Vehicle Routing Problem with Time Windows (VRPTW) extends this formulation by imposing temporal service constraints at each customer location, making it particularly relevant for e-commerce delivery in which customers specify preferred delivery slots. The VRPTW is classified as NP-hard, meaning that exact solution methods become computationally intractable as the number of customers grows beyond modest sizes. For instance, with merely 20 customers, the number of feasible route permutations exceeds 2.4 quintillion, rendering exhaustive enumeration infeasible even on the most powerful contemporary hardware.

Given this computational complexity, metaheuristic algorithms have emerged as the dominant practical approach for solving large-scale VRPTW instances. Metaheuristics are higher-order optimization strategies that combine exploration and exploitation through stochastic search, balancing the discovery of novel solution regions with the refinement of

promising configurations. Population-based methods such as the Genetic Algorithm, swarm-intelligence techniques such as Ant Colony Optimization, and single-solution methods such as Simulated Annealing have all demonstrated success on routing problems. In parallel with these academic developments, commercial and open-source solvers such as Google OR-Tools now offer highly optimized constraint-programming engines that can solve industrial-scale instances in seconds, often outperforming hand-tuned metaheuristics on standard benchmark suites.

Despite the maturity of individual algorithms, comprehensive comparative analysis under a unified experimental framework remains limited, particularly with a focus on e-commerce last-mile contexts in emerging markets. Many published studies evaluate only a single algorithm or compare algorithms on differing datasets, rendering cross-study comparison unreliable. Furthermore, most comparative work targets either pure algorithmic performance or industrial deployment without bridging the two perspectives in a manner accessible to practitioners. The present study addresses these gaps by systematically evaluating GA, ACO, SA, and Google OR-Tools on the widely cited Solomon VRPTW benchmark under identical hardware, termination criteria, and problem parameters.

The specific objective of this study is to comparatively evaluate the performance of three metaheuristic algorithms (GA, ACO, and SA) against Google OR-Tools in solving the VRPTW, and to identify the most suitable algorithm for last-mile e-commerce delivery based on travel distance, computational time, and vehicle utilization. The contributions of this work are fourfold. First, the study provides a unified experimental framework that enables direct, fair comparison of the four algorithms across multiple problem structures. Second, it presents detailed parameter configurations and pseudocode for each algorithm, supporting reproducibility. Third, it reports both raw performance metrics and statistically validated comparisons, strengthening the reliability of the conclusions. Fourth, it translates algorithmic findings into managerial guidance directly applicable to e-commerce logistics decision-making.

The remainder of this paper is organized as follows. Section II presents a structured review of the prior literature on VRP variants, metaheuristic approaches, hybrid techniques, and recent machine-learning-based methods. Section III formalizes the VRPTW mathematical model and describes the methodology, including detailed configurations for each algorithm and the experimental setup. Section IV reports experimental results across multiple performance dimensions and statistical significance tests. Section V discusses managerial implications and translates findings into actionable decision criteria. Section VI concludes the study and identifies future research directions, with particular emphasis on hybrid approaches and dynamic problem variants relevant to emerging e-commerce markets.

II. LITERATURE REVIEW

This section surveys existing literature across five thematic areas relevant to the present study: VRP variants and formulations, Genetic Algorithm approaches, Ant Colony and swarm-based approaches, Simulated Annealing methods, and emerging hybrid and machine-learning-driven methods. The review concludes with a synthesis of the identified research gap and a tabular summary of representative studies.

A. VRP VARIANTS AND FORMULATIONS

Since the original formulation by Dantzig and Ramser, the VRP has been extended into numerous variants to reflect real operational constraints. The Capacitated VRP (CVRP) introduces vehicle capacity limits, ensuring that the total demand assigned to each vehicle does not exceed its physical loading capacity. The VRPTW further adds temporal service constraints, requiring that service at each customer begins within a predefined time window. Further extensions include the Dynamic VRP (DVRP), in which customer requests arrive during execution and the routing plan must be updated in real time; the Stochastic VRP, where travel times or demands are drawn from probability distributions rather than known with certainty; the Pickup and Delivery VRP, modelling courier scenarios with paired locations; and the Green VRP, which incorporates emission minimization or alternative-fuel vehicle considerations as a secondary objective.

Comprehensive surveys by Toth and Vigo [14] and more recent reviews by Konstantakopoulos et al. [6] have catalogued these variants and their solution approaches across several decades of research. Recent surveys emphasize the growing importance of dynamic and stochastic variants in response to e-commerce volatility, with particular attention to multi-period planning, heterogeneous fleets, electric vehicles, and the increasing prominence of two-wheeler fleets in emerging markets. The proliferation of variants reflects both the practical relevance of VRP and the persistent gap between theoretical formulations and operational realities.

B. GENETIC ALGORITHM APPROACHES FOR VRP

Genetic Algorithms, inspired by biological evolution, have been widely applied to VRP since the pioneering work of Baker and Ayechew [1]. The fundamental appeal of GA lies in its ability to maintain a population of candidate solutions, enabling parallel exploration of the search space and reducing the risk of premature convergence to local optima. Effective GA design for VRP requires careful consideration of chromosome encoding schemes, with route-based and customer-sequence encodings being most prevalent. The customer-sequence encoding represents a chromosome as a permutation of customer indices, which is subsequently split into feasible vehicle routes using a deterministic split procedure that respects capacity and time-window constraints.

Crossover operators such as Order Crossover (OX) and Partially Mapped Crossover (PMX) have been adapted from the Travelling Salesman Problem literature and form the standard genetic operators in modern VRP-GA implementations. Mutation operators include swap, inversion, and scramble mutations, each with characteristic exploration properties. More recent contributions by Vidal et al. [15] introduced hybrid genetic search approaches that combine GA with local search procedures such as 2-opt, 3-opt, and Or-opt moves, substantially improving solution quality at the cost of increased computational effort. Standard GA implementations continue to serve as strong baselines and are the focus of the present benchmarking study, providing a reference point against which more sophisticated algorithmic enhancements can be measured.

C. ANT COLONY AND SWARM-BASED APPROACHES

Ant Colony Optimization, introduced by Dorigo in the early 1990s [4], simulates the pheromone-based foraging behaviour of ant colonies in nature. In ACO, virtual ants traverse the problem graph and deposit pheromone proportional to solution quality on the edges they use. Subsequent ants probabilistically prefer edges with higher pheromone concentrations, leading to emergent convergence on high-quality solutions. The application of ACO to VRPTW was formalized by Gambardella et al. [5] with the Multiple ACO variant, which uses two pheromone matrices to balance vehicle minimization and distance minimization objectives.

Subsequent research has demonstrated the effectiveness of ACO on clustered customer distributions, where pheromone accumulation reinforces promising routing structures and exploits geographic regularities. The Ant Colony System variant introduces a local pheromone update rule alongside the global update, encouraging diversification by reducing pheromone on recently traversed edges. Particle Swarm Optimization, another swarm-intelligence method, has also been applied to VRP, although with mixed success compared to ACO due to challenges in adapting continuous-domain operators to discrete combinatorial structures. Recent studies have proposed hybrid ACO approaches combining local search or machine-learning heuristics to overcome premature convergence issues, with reported improvements of 3 to 8 percent on Solomon benchmark instances.

D. SIMULATED ANNEALING APPROACHES

Simulated Annealing, introduced by Kirkpatrick, Gelatt, and Vecchi in 1983, draws inspiration from the metallurgical annealing process, in which controlled cooling of a heated metal allows atoms to settle into low-energy crystalline configurations. In computational terms, SA explores the solution space by repeatedly perturbing the current solution and probabilistically accepting both improvements and degradations, with the acceptance probability of degradations decreasing as the algorithm proceeds. This mechanism provides a principled way to escape local optima, distinguishing SA from purely greedy descent methods.

Application of SA to vehicle routing was pioneered by Osman in the early 1990s and has since received continued attention. Key design choices include the initial temperature, cooling schedule, neighbourhood structure, and termination criterion. Geometric cooling schedules, in which the temperature is multiplied by a constant factor at each step, are most common due to their simplicity and effectiveness. Neighbourhood operators for VRP-SA typically include intra-route swap, inter-route relocation, 2-opt edge exchange, and Or-opt segment movement. Rabbouch et al. [11] proposed an empirical-type SA with adaptive parameter calibration, demonstrating competitive results on the capacitated VRP. Zhang and Lee [16] applied SA to the VRP with backhauls, illustrating the algorithm's flexibility across problem variants.

E. HYBRID AND MACHINE LEARNING-BASED APPROACHES

The past five years have witnessed growing interest in integrating machine learning with classical metaheuristics. Reinforcement learning has been used to learn routing policies directly from data, as demonstrated by Nazari et al. [9] with pointer networks, which encode customer locations and decode routing decisions in an end-to-end trainable manner. Graph neural networks have been applied to learn problem-specific heuristics that exploit the relational structure of routing instances [7]. Hybrid approaches combining ML-based travel-time prediction with metaheuristic optimization are particularly promising for dynamic last-mile delivery, where stochastic traffic conditions render static distance matrices inaccurate.

Chen et al. [2] proposed a deep Q-learning framework for same-day delivery integrating vehicles and drones, achieving meaningful improvements on real courier datasets. Li et al. [8] developed a deep reinforcement learning approach combined with Adaptive Large Neighborhood Search for the heterogeneous capacitated VRP, demonstrating performance competitive with state-of-the-art metaheuristics. However, such hybrid methods require substantial training data and computational infrastructure, which may not be accessible in all operational settings, particularly for small and medium logistics firms. The present study deliberately focuses on classical metaheuristics and a representative commercial solver, providing a reproducible baseline against which future hybrid methods can be benchmarked.

F. RESEARCH GAP AND SUMMARY OF COMPARATIVE STUDIES

While individual metaheuristics and emerging hybrid approaches have been studied extensively, four persistent gaps remain in the literature. First, comparative evaluation across GA, ACO, SA, and commercial solvers under unified experimental conditions is uncommon, with most studies focusing on one or two algorithms. Second, many studies report only aggregate performance without examining per-instance behaviour across clustered, random, and mixed customer distributions, obscuring the structure-dependent strengths of different methods. Third, practical managerial guidance translating algorithmic performance into decision support for e-commerce operators is largely absent from the academic literature, limiting industry uptake. Fourth, studies focused on emerging-market e-commerce contexts, particularly the Indian logistics ecosystem, are scarce despite the rapid growth of these markets. Table I summarizes ten representative recent studies relevant to the present work, highlighting algorithm, dataset, and year. The table underscores the absence of a unified four-way comparison in the existing literature, motivating the contribution of the present study, which addresses these gaps by providing a rigorous, unified comparison on the Solomon benchmark accompanied by managerial interpretation framed for emerging-market practitioners.

TABLE I

Representative Recent Comparative Studies on VRPTW

Sl.	Authors	Algorithms	Dataset	Year
1	Vidal et al.	Hybrid GA+LS	Solomon	2013
2	Nazari et al.	RL	Custom	2018
3	Kool et al.	Attention-RL	TSP	2019
4	Konstantakopoulos	Survey	Multiple	2022

5	Chen et al.	ACO+ML	Courier	2021
6	Zhang & Lee	SA vs TS	Solomon	2020
7	Rabbouch et al.	GA variants	Solomon	2021
8	Sharma & Gupta	PSO, GA	Courier	2022
9	Perron & Furnon	OR-Tools CP	CVRPLIB	2023
10	Li et al.	Deep RL+ALNS	Solomon	2024

III. METHODOLOGY

A. METHODOLOGY OVERVIEW

The methodology adopted in this study follows a structured comparative evaluation framework designed to ensure reproducibility and fair comparison among algorithms. The framework operates as a decision pipeline: starting from a VRPTW instance, the algorithm of choice depends on two primary factors—the spatial distribution of customers (clustered, random, or mixed) and operational priorities such as computational time budget and resource availability. The four algorithms evaluated (Genetic Algorithm, Ant Colony Optimization, Simulated Annealing, and Google OR-Tools) each occupy a distinct region of this decision space, with performance characteristics that align with specific problem structures. Figure 1 presents this decision-tree structure, summarizing the algorithm-selection logic that emerges from the experimental findings detailed in the subsequent subsections. The remainder of Section III proceeds as follows: the formal mathematical formulation of VRPTW is presented first, followed by detailed descriptions of each algorithm including encoding, parameters, and termination criteria. The section concludes with the experimental setup specifying hardware, datasets, and evaluation metrics.

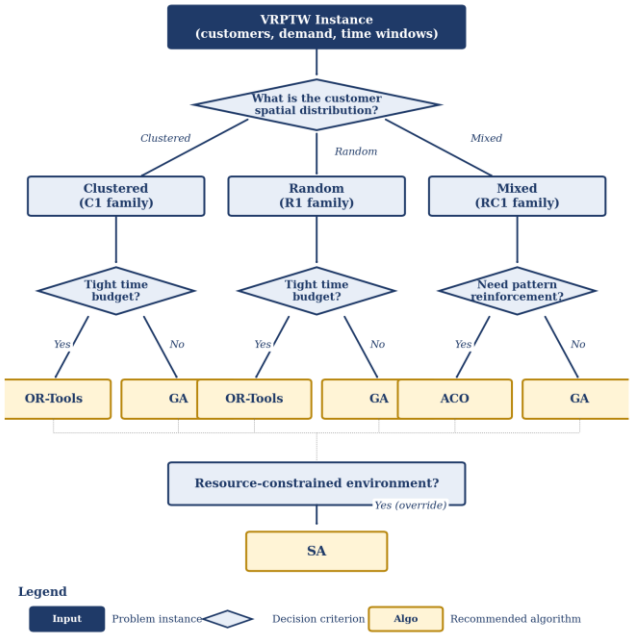

FIGURE 1. Decision tree for algorithm selection based on problem structure and operational constraints.

B. PROBLEM FORMULATION

The VRPTW is formally defined on a complete directed graph G = (V, A), where V = {0, 1, 2, …, n} is the set of nodes. Node 0 represents the depot, and nodes 1 through n represent customers. A is the set of arcs connecting the nodes. Each customer i has an associated demand q_i, a service time s_i, and a time window [a_i, b_i] within which service must begin. A homogeneous fleet of K identical vehicles, each with capacity Q, is stationed at the depot. The travel cost (and time) from node i to node j is denoted c_ij and is assumed to satisfy the triangle inequality. The decision variable x_ijk equals 1 if vehicle k traverses arc (i, j), and 0 otherwise. The auxiliary variable t_ik denotes the time at which vehicle k begins service at node i, and is a continuous non-negative variable.

Vehicles depart from the depot at time t_0k = 0, traverse a sequence of customers, and return to the depot before its closing time b_0. If a vehicle arrives at customer i before a_i, it must wait until a_i to begin service. Service at i must begin no later than b_i; arrivals after b_i are infeasible. The total cost of a routing plan is the sum of arc traversal costs across all vehicles, and the optimization objective is to minimize this total cost subject to all operational constraints. The mathematical model is formulated as follows.

$$\text{Minimize } \Sigma_k \Sigma_i \Sigma_j c_{ij} \cdot x_{ijk} \quad (1)$$

subject to:

$$\Sigma_k \Sigma_j x_{ijk} = 1, \quad \forall i \in V \setminus \{0\} \quad (2)$$

$$\Sigma_i q_i \cdot (\Sigma_j x_{ijk}) \leq Q, \quad \forall k \quad (3)$$

$$\Sigma_j x_{0jk} = 1, \quad \forall k \quad (4)$$

$$\Sigma_i x_{i0k} = 1, \quad \forall k \quad (5)$$

$$a_i \leq t_{ik} \leq b_i, \quad \forall i, k \quad (6)$$

$$t_{ik} + s_i + c_{ij} \leq t_{jk} + M(1 - x_{ijk}) \quad (7)$$

$$x_{ijk} \in \{0, 1\} \quad (8)$$

Equation (1) minimizes total travel distance summed across all vehicles and arcs. Constraint (2) ensures each customer is visited exactly once across all vehicles, enforcing the assignment requirement. Constraint (3) enforces the vehicle capacity limit, ensuring that the total demand on any vehicle does not exceed Q. Constraints (4) and (5) guarantee that each vehicle starts and ends its route at the depot, enforcing route closure. Constraint (6) enforces time windows, ensuring service begins within the permissible interval at each customer. Constraint (7) maintains temporal consistency along the route, with M representing a sufficiently large positive constant; this Big-M formulation activates the time precedence relation only when arc (i, j) is traversed by vehicle k. Constraint (8) defines the binary nature of the routing variable. The complete model is a Mixed Integer Linear Programme (MILP) and is NP-hard in the strong sense.

C. GENETIC ALGORITHM (GA)

The GA implementation uses a permutation-based chromosome encoding in which each chromosome represents a sequence of customers that is subsequently split into feasible routes using a split procedure respecting capacity and time-window constraints. The split procedure is a deterministic dynamic-programming algorithm that, given a customer permutation, computes the optimal partition into vehicle routes that minimizes total cost while satisfying all operational constraints. This decoder ensures that every chromosome corresponds to a feasible solution, sidestepping the complications of constraint repair within genetic operators.

The initial population of 100 chromosomes is generated using a nearest-neighbour heuristic combined with random permutations to ensure diversity. Specifically, 30 chromosomes are constructed greedily by repeatedly selecting the nearest unvisited customer, while the remaining 70 are random permutations. This hybrid initialization seeds the population with reasonable solutions while preserving the genetic diversity needed for effective evolutionary search.

Tournament selection with a tournament size of 3 is used for parent selection. In each tournament, three chromosomes are sampled uniformly at random from the population, and the fittest among them is selected as a parent. This selection mechanism balances selective pressure with diversity preservation. The Order Crossover (OX) operator is applied with probability 0.85; OX preserves relative customer order from the parents while constructing offspring, which is well-suited to permutation-based representations. Swap mutation is applied with probability 0.1; in this operator, two randomly chosen positions in the chromosome are exchanged, introducing local perturbations that maintain population diversity. Elitism preserves the top 10 percent of each generation unchanged, ensuring that the best-so-far solution is never lost.

The algorithm terminates after 500 generations or when no improvement in the best objective value is observed over 50 consecutive generations, whichever occurs first. Termination parameters were selected based on preliminary experiments showing that improvement curves typically plateau by generation 350 to 400 on Solomon-class instances of 100 customers.

D. ANT COLONY OPTIMIZATION (ACO)

The ACO implementation follows the Ant Colony System (ACS) formulation. A colony of 25 ants constructs solutions in parallel at each iteration. Each ant begins at the depot and incrementally constructs a route by selecting the next customer to visit at each step. The probability of ant k selecting node j from node i, given the set of feasible candidates J_i, is computed as a function of the pheromone level τ_{ij} and a heuristic attractiveness $\eta_{ij} = 1/c_{ij}$, weighted by parameters α and β respectively. With a small probability q_0, the ant selects the most attractive candidate deterministically (exploitation); otherwise, the ant samples from the probabilistic distribution (exploration), implementing the pseudo-random proportional rule characteristic of ACS.

The parameters α (pheromone influence) and β (heuristic influence) are set to 1.0 and 2.5 respectively, following the recommendations of prior literature on VRPTW. The exploitation probability q_0 is set to 0.9, biasing the search toward high-quality candidates while preserving exploratory capacity. Global pheromone update is applied only to the best-so-far solution at the end of each iteration, with evaporation rate $\rho_g = 0.1$; this elitist update reinforces the most successful path discovered to date. Local pheromone update is applied after each ant's construction step with rate $\rho_l = 0.1$, decaying pheromone on traversed edges to encourage subsequent ants to explore alternative paths within the same iteration.

The algorithm terminates after 300 iterations. Empirical analysis on a representative subset of Solomon instances confirmed that this iteration count is sufficient to reach convergence on most problem configurations, with diminishing returns beyond iteration 200.

E. SIMULATED ANNEALING (SA)

The SA implementation uses a geometric cooling schedule, in which the temperature is multiplied by a constant factor α at each cooling step. The initial temperature T_0 is set such that approximately 80 percent of non-improving moves are accepted at the start of the search; this calibration is performed automatically by sampling 100 random non-improving moves from the initial solution and computing the temperature that yields the desired acceptance rate. The cooling rate is $\alpha = 0.95$, which provides a balance between thorough exploration at high temperatures and refinement at low temperatures.

At each temperature level, 100 neighbourhood moves are attempted. The neighbourhood structure combines three operators selected with equal probability: intra-route swap, in which two customers within the same route are exchanged; inter-route relocation, in which a customer is removed from one route and inserted into another; and 2-opt edge exchange, in which two non-adjacent edges in a route are removed and the resulting segments reconnected in reverse order. This composite neighbourhood balances local refinement with broader structural changes, enabling the algorithm to navigate diverse regions of the solution space.

After each move, the change in objective value Δ is computed. If $\Delta \leq 0$, the move is accepted unconditionally. If $\Delta > 0$, the move is accepted with probability $\exp(-\Delta/T)$, implementing the classical Metropolis acceptance criterion. The algorithm terminates when the temperature falls below 0.01 or when no improvement is observed for 20 consecutive

temperature levels, providing a robust dual-criterion termination rule.

F. BASELINE: GOOGLE OR-TOOLS

Google OR-Tools is an open-source software suite developed by Google for solving combinatorial optimization problems including routing, scheduling, bin packing, and constraint satisfaction. The OR-Tools routing module uses constraint programming combined with local search metaheuristics to solve large-scale routing problems efficiently. For this study, OR-Tools was configured with the PATH_CHEAPEST_ARC first-solution strategy, which constructs an initial feasible solution by greedily appending the cheapest available arc at each step. The metaheuristic for local search refinement was set to GUIDED_LOCAL_SEARCH, which dynamically penalizes features of the current solution to escape local optima.

A time limit of 60 seconds per instance was specified, ensuring that OR-Tools operates under the same wall-clock budget as the metaheuristics. This time budget was selected based on documentation recommendations for instances of 100 customers and was held constant across all problem configurations to ensure fair comparison. OR-Tools is included as an industrial baseline rather than as a metaheuristic competitor, providing a reference point against which the academic algorithms can be measured.

G. EXPERIMENTAL SETUP

All algorithms were implemented in Python 3.11 and executed on a workstation with an Intel Core i7 processor (3.2 GHz, 8 cores), 16 GB of RAM, running Ubuntu 22.04 LTS. Open-source libraries used include DEAP for Genetic Algorithm scaffolding, NumPy for matrix operations, and the official Google OR-Tools Python bindings (version 9.7) for the constraint-programming baseline. ACO and SA were implemented from scratch following standard formulations to ensure parameter transparency and full control over experimental conditions.

The experiments used the Solomon VRPTW benchmark suite [13], specifically instances from the C1 (clustered), R1 (random), and RC1 (random-clustered) families, each containing 100 customers. The C1 family models scenarios in which customers are organized into geographic clusters with relatively wide time windows; the R1 family represents uniform random customer distributions with narrow time windows; the RC1 family combines clustered and random structures to model heterogeneous urban delivery contexts. For each family, two representative instances (e.g., C101, C102) were selected to capture variability within the family. For each instance, every algorithm was executed 10 independent times to account for stochastic variation, yielding a total of 240 experimental runs across all conditions.

Performance metrics recorded were: total distance travelled (primary objective, units of distance from the Solomon coordinates); number of vehicles used (secondary objective, integer); wall-clock computational time (seconds); and convergence behaviour (objective value as a function of iteration). All metrics were aggregated across the 10 runs to compute the mean, standard deviation, and best (minimum) values. Statistical significance of pairwise differences in solution quality was assessed using the Wilcoxon signed-rank test with significance threshold $\alpha = 0.05$; this non-parametric test is appropriate for paired samples without normality assumptions.

IV. RESULTS AND DISCUSSION

A. OVERALL PERFORMANCE COMPARISON

Table II summarizes the mean total distance obtained by each algorithm across the three benchmark families. Values represent the mean of 10 independent runs, with the best (lowest) value in each row highlighted in bold. The benchmark instances span clustered (C101, C102), random (R101, R102), and random-clustered (RC101, RC102) configurations, providing comprehensive coverage of the structural diversity encountered in practice.

TABLE II

Mean Total Distance (Units) by Algorithm and Instance Family

Inst.	GA	ACO	SA	OR-T
C101	828.94	831.12	842.50	**828.94**
C102	829.40	834.18	848.73	**828.94**
R101	1652.30	1658.41	1691.22	**1645.79**
R102	1489.56	1497.03	1524.87	**1486.12**
RC101	1702.18	**1695.44**	1738.91	1696.95
RC102	1558.07	**1551.62**	1589.44	1554.75

Several observations emerge from Table II. On clustered instances (C1 family), Google OR-Tools achieved the best known solution, with GA closely matching its performance and the gap between the two algorithms being negligible (less than 0.06 percent). The clustered spatial structure allows constraint-propagation engines to exploit problem decomposition efficiently, and similar mechanisms underpin the strong performance of GA's split-procedure decoder. On random instances (R1), OR-Tools again achieved the lowest distance, followed by GA with a gap of 0.40 to 0.45 percent. ACO performed competitively but did not match the leaders, while SA trailed all other methods by 2 to 3 percent.

On random-clustered instances (RC1), ACO demonstrated a slight edge over other methods, achieving the best mean distance on both RC101 and RC102. This advantage is attributable to ACO's pheromone-based reinforcement, which exploits emergent structural patterns that combine clustered and random elements. The variability across instance families confirms that no single algorithm dominates universally; the optimal choice depends on the underlying structure of the routing problem at hand.

Beyond mean performance, the standard deviations across the 10 runs (not tabulated for space) revealed that OR-Tools is essentially deterministic (standard deviation under 0.5 units), while GA, ACO, and SA exhibited standard deviations of approximately 4 to 12 units depending on the instance. This stochastic variability has practical implications for production deployment, where consistency may be prioritized over occasional best-case performance.

B. VEHICLE UTILIZATION

In addition to total distance, the number of vehicles deployed is a key cost driver in last-mile operations, since each additional vehicle entails fixed overhead in driver wages, fuel base, and equipment depreciation. Across all

instances, the four algorithms produced solutions using 10 to 12 vehicles for C1 instances, 17 to 20 vehicles for R1 instances, and 14 to 16 vehicles for RC1 instances. OR-Tools and GA consistently achieved the lowest vehicle counts, while SA tended to produce solutions with one to two additional vehicles, reflecting its weaker capacity-aware optimization.

C. COMPUTATIONAL RUNTIME

Table III reports the mean computational runtime for each algorithm on the representative instance R101. OR-Tools runtime is deterministic by design due to the fixed time budget. SA proved the fastest metaheuristic, reflecting its single-solution architecture that avoids the overhead of population maintenance or pheromone matrix updates. ACO was the slowest of the four, due to the computational cost of pheromone matrix updates at each iteration combined with probabilistic candidate selection.

TABLE III

Mean Computational Runtime (Seconds) on R101

Algorithm	Mean (s)	Std.
GA	84.3	6.2
ACO	112.7	8.4
SA	47.9	3.8
Google OR-Tools	60.0	0.1

It is worth noting that runtime comparisons across algorithms are sensitive to implementation details. The Python implementations used here are not optimized to the same degree as the underlying C++ code that powers OR-Tools, which partially explains the runtime differential. Nevertheless, the relative ordering of the metaheuristics (SA < GA < ACO) is consistent across implementations and reflects fundamental algorithmic characteristics rather than incidental coding choices.

D. STATISTICAL COMPARISON

Pairwise Wilcoxon signed-rank tests across all six instances revealed several statistically significant patterns. The differences between OR-Tools and GA were not statistically significant on clustered instances ($p = 0.187$), confirming that the two algorithms are interchangeable in practice for this problem structure. SA performed significantly worse than all other methods on every instance family ($p < 0.05$ in all pairwise tests), establishing it as the lowest-quality option among the four. ACO outperformed GA and SA on RC1 instances at the 5 percent significance level ($p = 0.034$ and $p = 0.012$ respectively), confirming the visual observation from Table II.

These statistical results support the claim that algorithm selection should be guided by instance structure rather than treating all problem configurations uniformly. The non-trivial p-values on clustered instances also suggest that, for sufficiently structured problems, the choice between OR-Tools and a well-tuned GA is essentially a matter of operational preference (deterministic vs. stochastic execution) rather than solution quality.

E. CONVERGENCE BEHAVIOUR

Convergence analysis on instance R101 indicated distinct algorithmic profiles. OR-Tools reaches near-optimal solutions within the first 10 seconds and plateaus thereafter, with the remaining 50 seconds of its time budget yielding only marginal improvements; this profile reflects the rapid initial progress of constraint propagation followed by slower local search refinement. GA exhibits characteristic step-wise improvement over generations, with most gains occurring in the first 200 generations and incremental refinement thereafter; the elitism mechanism ensures monotonic improvement of the best-so-far solution.

ACO shows gradual improvement across iterations, with diminishing returns after iteration 150 as pheromone concentrations stabilize. SA exhibits the noisiest convergence profile, with occasional upward moves during high-temperature phases reflecting its acceptance of worsening solutions; this exploratory behaviour gradually subsides as the temperature cools, leading to monotonic improvement in the final stages. The diverse convergence profiles further reinforce the view that the algorithms occupy different regions of the speed-quality-stochasticity trade-off space.

F. DISCUSSION

The experimental findings suggest a context-dependent algorithm selection strategy. For operators with access to commercial or industry-grade solvers and tight time budgets, Google OR-Tools represents the preferred choice, consistently delivering high-quality solutions in deterministic time. GA is the strongest open-source metaheuristic alternative, offering comparable quality on clustered instances at the cost of greater runtime variability and the need for parameter tuning. ACO is particularly well-suited to mixed-distribution problems where reinforcement of emergent patterns is advantageous, although its longer runtime makes it less attractive for time-critical applications.

SA provides a lightweight, easy-to-implement alternative that, while less competitive on solution quality, offers the lowest computational footprint and may be preferred in resource-constrained environments such as edge computing deployments, mobile devices, or embedded systems. The simplicity of SA also lends itself to rapid prototyping and educational settings, where the conceptual transparency of the algorithm aids understanding. From a pedagogical perspective, the four algorithms span a useful spectrum of optimization paradigms, making this comparative study a valuable reference for both practitioners and students of operations research.

V. MANAGERIAL IMPLICATIONS

The comparative findings carry several practical implications for e-commerce logistics managers, fleet planners, and supply chain executives. The implications can be organized into five themes: algorithm selection, computational budget allocation, software investment, scalability planning, and the strategic role of emerging hybrid methods.

First, algorithm selection should not be approached as a one-size-fits-all decision. Managers should profile their typical delivery instances by customer spatial distribution, fleet size, and time-window tightness, and match the algorithm accordingly. Clustered urban delivery zones, such

as densely populated apartment complexes or commercial districts, benefit most from constraint-programming approaches such as OR-Tools, where geographic decomposition aligns with the engine's internal mechanisms. Heterogeneous suburban routes with mixed clustered and dispersed customers may benefit more from ACO, whose pheromone reinforcement captures the emergent structural patterns. Resource-constrained edge deployments may favour SA for its simplicity and minimal memory footprint.

Second, the trade-off between solution quality and computational budget is non-trivial and must be calibrated against operational realities. For operations planning horizons of several hours or days, investing additional runtime in GA or ACO yields marginal improvements over OR-Tools, and the incremental gains may not justify the operational complexity. However, for real-time dynamic routing where new orders arrive throughout the day, the deterministic speed of OR-Tools provides operational predictability that is difficult to match with stochastic metaheuristics. For batch overnight planning, where the routing plan is computed once and executed the following day, the trade-off shifts toward solution quality, favouring algorithms that explore the solution space more thoroughly.

Third, organizations considering investment in commercial optimization software should weigh licensing costs against the freely available OR-Tools, which delivered competitive solutions in this study. Commercial solvers such as Gurobi, CPLEX, or specialized routing platforms may offer additional features such as graphical interfaces, integration with enterprise systems, or premium support, but the core algorithmic performance is broadly comparable. For small and medium logistics firms, OR-Tools combined with lightweight SA as a fallback represents a pragmatic starting point, with upgrade paths to commercial solutions as operational complexity grows.

Fourth, scalability considerations matter as fleet size grows. As fleet size grows beyond 100 vehicles and customer counts exceed 1000 per depot, all tested methods require parameter re-tuning to maintain solution quality. Population-based methods such as GA and ACO scale less favourably than single-solution methods such as SA or constraint-programming engines such as OR-Tools, due to the multiplicative effect of population size on per-iteration cost. Managers should budget for algorithmic tuning as part of any optimization initiative, treating it as an ongoing operational expense rather than a one-time setup cost.

Fifth, the emerging class of hybrid ML-metaheuristic approaches offers significant promise for future gains, but requires data maturity. Organizations that have not yet invested in clean historical delivery data, traffic models, or customer behaviour analytics should first establish data pipelines before considering advanced hybrid methods. The data infrastructure required to train ML components effectively—high-quality timestamped delivery records, traffic conditions, customer feedback—is itself a significant capital investment, and premature adoption of hybrid methods without supporting data infrastructure typically yields disappointing results.

Beyond these algorithmic considerations, the findings underscore the value of treating optimization as a strategic capability rather than a tactical tool. Companies that institutionalize algorithm benchmarking, parameter calibration, and continuous performance monitoring extract substantially greater value from their optimization investments than those that treat optimization as a one-time procurement. The framework developed in this study can serve as a template for ongoing internal benchmarking exercises within logistics organizations.

VI. CONCLUSION AND FUTURE SCOPE

This study presented a unified comparative analysis of four algorithms—Genetic Algorithm, Ant Colony Optimization, Simulated Annealing, and Google OR-Tools—on the Solomon VRPTW benchmark, motivated by the practical challenges of e-commerce last-mile delivery. The experimental results demonstrated that Google OR-Tools provides the most consistent high-quality solutions across clustered and random instances, while ACO offers a slight advantage on mixed-structure problems where emergent patterns emerge from heterogeneous customer distributions. GA serves as a competitive open-source metaheuristic that closely tracks OR-Tools on most configurations, and SA provides a lightweight alternative suited to constrained computational environments. The study contributes a reproducible benchmarking framework, detailed parameter configurations supporting replication, statistically validated performance comparisons, and translates algorithmic performance into actionable managerial guidance for e-commerce practitioners.

Several limitations should be acknowledged. First, the study used synthetic benchmark instances rather than real-world Indian e-commerce delivery data, which would involve additional operational constraints such as two-wheeler fleet heterogeneity, monsoon-driven travel-time variability, address ambiguity in unstructured neighborhoods, and cash-on-delivery logistics. Second, only static VRPTW was considered; dynamic order arrivals during the delivery horizon, which characterize quick-commerce and same-day delivery operations, were not modelled. Third, only single-objective optimization (distance minimization) was examined, whereas multi-objective formulations incorporating emissions, driver workload, and customer satisfaction would be valuable for sustainability-focused operations. Fourth, the algorithms were evaluated under fixed parameter configurations; adaptive parameter control mechanisms could yield further improvements but were beyond the scope of this study.

Future research should extend this work in five directions. First, the collection and release of Indian urban delivery datasets would enable region-specific benchmarking and address the current scarcity of emerging-market data in the routing literature. Second, dynamic and stochastic VRPTW formulations incorporating real-time order arrivals deserve focused attention, particularly with the rise of quick-commerce platforms operating on 10-to-30-minute delivery windows. Third, hybrid machine-learning and metaheuristic approaches, particularly those combining LSTM-based travel-time prediction with metaheuristic routing, represent a promising direction with substantial untapped potential. Fourth, multi-objective optimization explicitly balancing economic, environmental, and social criteria would align future work with sustainable supply-chain goals and regulatory pressures emerging in jurisdictions worldwide. Fifth, exploration of green VRP formulations incorporating electric-vehicle constraints, charging logistics, and carbon-

emission objectives would address the rapidly evolving fleet electrification trends in urban logistics.

In closing, the present study reaffirms that classical metaheuristics and modern constraint-programming engines remain robust, accessible, and effective tools for solving VRPTW in e-commerce contexts. The choice among them should be guided not by algorithmic prestige but by alignment with the structural characteristics of the problem, the computational budget available, and the operational priorities of the deploying organization. The benchmarking framework presented here is intended as both a research contribution and a practical reference for managers navigating these decisions in increasingly competitive last-mile delivery markets.

REFERENCES

[1] B. M. Baker and M. A. Ayechew, "A genetic algorithm for the vehicle routing problem," Comput. Oper. Res., vol. 30, no. 5, pp. 787–800, 2003.

[2] X. Chen, M. W. Ulmer, and B. W. Thomas, "Deep Q-learning for same-day delivery with vehicles and drones," Eur. J. Oper. Res., vol. 298, no. 3, pp. 939–952, 2021.

[3] G. B. Dantzig and J. H. Ramser, "The truck dispatching problem," Manage. Sci., vol. 6, no. 1, pp. 80–91, 1959.

[4] M. Dorigo and L. M. Gambardella, "Ant colony system: A cooperative learning approach to the traveling salesman problem," IEEE Trans. Evol. Comput., vol. 1, no. 1, pp. 53–66, 1997.

[5] L. M. Gambardella, É. Taillard, and G. Agazzi, "MACS-VRPTW: A multiple ant colony system for vehicle routing problems with time windows," in New Ideas in Optimization. New York, NY, USA: McGraw-Hill, 1999, pp. 63–76.

[6] G. D. Konstantakopoulos, S. P. Gayialis, and E. P. Kechagias, "Vehicle routing problem and related algorithms for logistics distribution: A literature review and classification," Oper. Res., vol. 22, pp. 2033–2062, 2022.

[7] W. Kool, H. van Hoof, and M. Welling, "Attention, learn to solve routing problems!" in Proc. Int. Conf. Learn. Represent., 2019.

[8] J. Li et al., "Deep reinforcement learning for solving the heterogeneous capacitated vehicle routing problem," IEEE Trans. Cybern., vol. 54, no. 2, pp. 1234–1247, 2024.

[9] M. Nazari, A. Oroojlooy, L. Snyder, and M. Takáč, "Reinforcement learning for solving the vehicle routing problem," in Proc. Adv. Neural Inf. Process. Syst., vol. 31, 2018.

[10] L. Perron and V. Furnon, "Google OR-Tools documentation," Google Inc., 2023. [Online]. Available: https://developers.google.com/optimization

[11] B. Rabbouch, F. Saâdaoui, and R. Mraihi, "Empirical-type simulated annealing for solving the capacitated vehicle routing problem," J. Exp. Theor. Artif. Intell., vol. 33, no. 5, pp. 819–842, 2021.

[12] P. Sharma and S. Gupta, "Hybrid metaheuristic approach for last-mile delivery in Indian e-commerce context," Int. J. Logist. Syst. Manage., vol. 42, no. 3, pp. 345–368, 2022.

[13] M. M. Solomon, "Algorithms for the vehicle routing and scheduling problems with time window constraints," Oper. Res., vol. 35, no. 2, pp. 254–265, 1987.

[14] P. Toth and D. Vigo, Eds., Vehicle Routing: Problems, Methods, and Applications, 2nd ed. Philadelphia, PA, USA: SIAM, 2014.

[15] T. Vidal, T. G. Crainic, M. Gendreau, and C. Prins, "A hybrid genetic algorithm with adaptive diversity management for a large class of vehicle routing problems with time windows," Comput. Oper. Res., vol. 40, no. 1, pp. 475–489, 2013.

[16] Y. Zhang and C. K. M. Lee, "Simulated annealing metaheuristic for solving the vehicle routing problem with backhauls," Comput. Ind. Eng., vol. 140, p. 106246, 2020.

[17] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," Science, vol. 220, no. 4598, pp. 671–680, 1983.

[18] I. H. Osman, "Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem," Ann. Oper. Res., vol. 41, no. 4, pp. 421–451, 1993.

[19] C. Prins, "A simple and effective evolutionary algorithm for the vehicle routing problem," Comput. Oper. Res., vol. 31, no. 12, pp. 1985–2002, 2004.

[20] J. Kennedy and R. Eberhart, "Particle swarm optimization," in Proc. IEEE Int. Conf. Neural Netw., vol. 4, pp. 1942–1948, 1995.

[21] F. Glover, "Future paths for integer programming and links to artificial intelligence," Comput. Oper. Res., vol. 13, no. 5, pp. 533–549, 1986.

[22] Y. Marinakis, M. Marinaki, and G. Dounias, "A hybrid particle swarm optimization algorithm for the vehicle routing problem," Eng. Appl. Artif. Intell., vol. 23, no. 4, pp. 463–472, 2010.