Ingénierie des Systèmes d'Information
Vol. 31, No. 8, August, 2026, pp. 2569-2578
Journal homepage: http://iieta.org/journals/isi

An Edge-Aware Adaptive Hybrid Discrete Firefly Algorithm for UAV-Assisted IoT Data-Collection Route Optimization

Meaad Mohammed Salih[1*], Raya Basil Alothman[1], Ali A. Al-Arbo[2]

[1] Department of Computer Science, College of Education for Pure Science, University of Mosul, Mosul 41002, Iraq
[2] Department of English Language, College of Arts, University of Mosul, Mosul 41002, Iraq

Corresponding Author Email: meaad_mahammed@uomosul.edu.iq

https://doi.org/10.18280/isi.310804

Received: 14 May 2026
Revised: 29 July 2026
Accepted: 11 August 2026
Available online: 31 August 2026

Keywords:
Adaptive Hybrid Discrete Firefly Algorithm, edge-aware search, metaheuristic optimization, route optimization, travelling salesman problem, Two-Opt local refinement, UAV-assisted IoT

ABSTRACT

Unmanned aerial vehicle (UAV)-assisted Internet of Things (IoT) data collection requires efficient route optimization to reduce travel cost and improve mission efficiency. Because the underlying routing task can be formulated as a discrete travelling salesman problem (TSP), effective metaheuristic design must preserve meaningful route structures throughout the search. This study proposes an Edge-Aware Adaptive Hybrid Discrete Firefly Algorithm (AHDFA) for UAV-assisted IoT data-collection route optimization. The method replaces conventional position-based similarity with an edge-overlap distance measure and combines guided swap, insertion, and inversion operators with stagnation-aware exploration and bounded Two-Opt refinement. AHDFA was evaluated on 15 Traveling Salesman Problem Library (TSPLIB) instances with 51–150 cities and 10 synthetic UAV-assisted IoT scenarios. It achieved the second-best overall rank among six methods and a mean relative percentage deviation (RPD) of 3.40% on the TSPLIB benchmarks. Holm-corrected statistical tests showed that AHDFA significantly outperformed several comparator methods, whereas differences from the stronger local-search-based comparators were not significant. These findings indicate that edge-aware search provides an effective and interpretable approach to UAV-assisted IoT route optimization, although validation on larger-scale and field-derived scenarios remains necessary.

1. INTRODUCTION

Unmanned aerial vehicle (UAV)-assisted data collection in clustered Internet of Things (IoT) networks requires a flight plan that visits multiple cluster heads, returns to the depot, and satisfies energy, latency, and service constraints. When the collection points are fixed for a single sortie, the routing core can be modeled as a travelling salesman problem (TSP). Because the TSP is NP-hard, practical planners rely on heuristics and metaheuristics. Edge-exchange local search, including Two-Opt and Lin-Kernighan, remains useful for removing crossing edges and refining candidate tours [1, 2]. This abstraction is consistent with UAV-assisted IoT studies in which trajectory length, service latency, energy consumption, and data freshness are coupled design factors [3-6]. The Firefly Algorithm is attractive because it provides a simple population-based attraction mechanism with a low implementation burden [7, 8]. However, directly applying the original continuous Firefly Algorithm to a permutation search space is not straightforward. Reviews emphasize that discrete performance depends mainly on how the distance between candidate permutations is measured and translated into feasible edits [9-11]. Representative discrete Firefly algorithms for the TSP have used swap-based distance and movement to adapt attraction to permutation problems [12, 13]. However, these formulations may classify shifted tours as highly dissimilar even when the tours share the same route edges. Recent reviews in routing and scheduling also emphasize hybridization, adaptive parameter control, and statistically robust evaluation [14, 15].

The proposed Adaptive Hybrid Discrete Firefly Algorithm (AHDFA) addresses these limitations through four linked design choices. First, edge-overlap distance replaces positional similarity so that attraction is based on preserved route adjacencies rather than arbitrary city indices. Second, guided swap, insertion and inversion operators allow a dimmer firefly to generate edges from a brighter firefly and keep the permutation feasible. Third, stagnation-aware reheating improves diversification only when the search is no longer improving and not according to a high randomness schedule. Fourth, bounded Two-Opt refinement eliminates local edge inefficiencies after guided movement, but not an exhaustive local search. AHDFA is therefore presented as a competitive and interpretable hybrid discrete Firefly framework for UAV-assisted IoT route optimization, not as a universal best TSP solver. Its contributions are fourfold: (i) an edge-aware attraction model with route structure; (ii) a guided multi-operator movement pipeline with adaptive reheating; (iii) a UAV-assisted IoT routing formulation with arrival time-based freshness extension; and (iv) a larger experimental study with 15 TSPLIB cases, 30 seeds per stochastic algorithm, 6 comparator methods, multi-instance ablation study, and a 10-

scenario UAV-assisted IoT routing family. The rest of the paper is organized as follows. Section 2 reviews related work. Section 3 presents the routing model and proposed algorithm. Section 4 describes the experimental setup. Section 5 offers benchmark and case study results. Section 6 concludes the paper.

2. RELATED WORK

The classical TSP research showed that local edge exchanges can close a large fraction of the optimality gap with low computational cost. Croes introduced the Two-Opt improvement strategy [1]. Lin and Kernighan followed this approach by variable-depth exchanges and this approach is still a good reference for tour improvement [2]. The metaheuristic still combines local search and global exploration [14, 15]. The main issue is the interplay between global search and edge repair. The problem is that if local search is applied only as a final polishing step, tours may remain poorly organized in the population. If it is applied too aggressively, then the computational costs may take over. So a bounded local search rather than a full neighborhood scan is the best choice at every step. The Firefly algorithm treats each solution of the system as a light source that is less attractive to travel on with distance [7, 8]. In continuous spaces, the distance is usually geometric. In permutation spaces, it could be identified by position mismatch, swap sequence length, insertion cost, inversion distance, or edge overlap. Position-based measures are simple, but can say that the equivalent cyclic tours are dissimilar when the same set of edges is shifted to different indices. Swap sequence measures, based on some of the best discrete Firefly studies of the TSP [12, 13], allow for actionable edits, but do not directly measure the adjacencies that determine tour length. Edge overlap distance is more closely linked to the structure of the TSP, since it is comparing the edges in the closed tour. Movement operators also are different, too: swaps are simple but disruptive, insertions move a city but keep more of the sequence and inversions naturally fix crossings. Adaptive mechanisms can include a static randomization schedule or feedback-based reheating. AHDFA combines these features by using edge overlap to select meaningful targets, guided edits to create target adjacencies, and stagnation-triggered reheating to restore exploration only when needed.

In UAV applications, data collection has received sustained attention because UAVs can extend coverage, reduce dependence on fixed infrastructure, and support delay-sensitive sensing missions [3]. Systematic reviews and foundational age-of-information studies show that route design, service order, and hovering time jointly affect data freshness at the sink [4, 16-18]. Recent UAV-assisted IoT and multi-UAV studies have extended this work to emergency resource scheduling and path planning, dynamic task assignment, and task scheduling in IoT-enabled UAV edge/fog environments [19-21]. Related studies in swarm robotics, transportation optimization, and unconstrained optimization also support adaptive global search combined with local refinement [22-24]. Collectively, these studies indicate that route-planning algorithms should remain robust under changing mission constraints rather than perform well only on static benchmark tours.

Existing discrete Firefly approaches leave three issues only partially resolved for UAV-assisted TSP routing. First, distance definitions are often based on position or edit count rather than route edges, although adjacencies determine tour cost. Second, adaptive control is often scheduled in advance instead of being triggered by stagnation. Third, local refinement is often separated from population movement, which makes global attraction and edge-level improvement less compatible. AHDFA fills this gap by aligning attraction with edge overlap, applying guided feasible movement operators and embedding a bounded Two-Opt stage in the iterative search pipeline.

3. PROBLEM FORMULATION AND PROPOSED METHOD

3.1 UAV-assisted IoT routing model

We consider a set of collection nodes $V = \{0, 1, \ldots, n\}$, where node 0 is the depot and nodes 1, …, n are UAV-visited cluster heads. Let $d(i, j)$ be the travel cost between nodes i and j. We denote a candidate route by a permutation $\pi = (\pi(1), \ldots, \pi(n))$ of the non-depot nodes. The distance objective is:

$$f(\pi) = d(0, \pi(1)) + \sum_{k=1}^{n-1} d\,(\pi(k), \pi(k+1)) + d(\pi(n), 0) \tag{1}$$

where, the UAV leaves the depot, visits each cluster head exactly once and returns to the depot. So, with UAV-assisted IoT data collection, minimizing this closed tour is a practical approach for reducing total flight time and propulsion energy [3]. Figure 1 illustrates the scenario that we will be considering here: ground sensors move forward to the cluster heads of the sensors, a UAV visits the cluster heads once to download buffered data and then it returns to the depot or edge-cloud gateway. A route-planning module that takes only the shortest distance is usually enough when the data have the same urgency. But time-sensitive sensing applications also care about information freshness. Let $t(i, j) = d(i, j)/v$ be the travel time at cruise speed v, and let $s(i)$ be the service time at node i. The arrival time at the k-th visited node can be written as:

$$A_{\pi(k)} = \sum_{h=0}^{k-1} t\left(\pi(h), \pi(h+1)\right) + \sum_{h=1}^{k-1} s\left(\pi(h)\right), \quad k = 1, \ldots, n \tag{2}$$

where, $\pi(0) = 0$.

A weighted freshness proxy based on arrival times is:

$$g(\pi) = \sum_{k=1}^{n} w_{\pi(k)}\, A_{\pi(k)} \tag{3}$$

and a composite distance-freshness objective is:

$$F(\pi) = f(\pi) + \lambda g(\pi) \tag{4}$$

where, $w_{\pi(k)}$ is a priority weight and λ controls the trade-off between travel cost and freshness. The experiments focused on the distance-only objective in Eq. (1). Eqs. (2)–(4) provide a direct extension to age-of-information-aware mission planning, consistent with recent UAV-assisted IoT studies [4, 16-18].

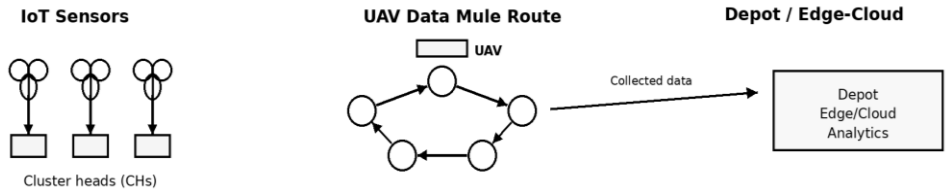

Figure 1. UAV-assisted IoT data-collection architecture showing ground sensors, cluster heads, the UAV closed tour, and depot/edge-cloud processing

3.2 Edge-overlap attraction and guided movement

In the AHDFA implementation, each firefly was encoded as a feasible tour. Because shorter tours were preferred, firefly brightness was defined by a monotone transformation of the objective value:

$$I(\pi) = \frac{1}{1 + f(\pi)} \tag{5}$$

The central design choice was the distance metric used to compare two tours. Rather than comparing city positions, AHDFA compared undirected edges because high-quality TSP tours are primarily distinguished by the adjacencies they preserve. Let $E(\pi)$ denote the set of edges in tour π. The edge-overlap distance between two fireflies i and j was defined as:

$$\delta(\pi_i, \pi_j) = n - \left|E(\pi_i) \cap E(\pi_j)\right| \tag{6}$$

A small value of $\delta(\pi_i,\pi_j)$ indicates that the tours share many edges, even if the same cities appear at different permutation indices. For example, tours A = (1, 2, 3, 4, 5, 6) and B = (3, 4, 5, 6, 1, 2) place no city at the same index. Nevertheless, they contain the same undirected cycle edges. A position-based metric would therefore overstate their dissimilarity, whereas δ(A, B) = 0. By contrast, C = (1, 3, 2, 4, 5, 6) breaks several adjacencies, so its edge-overlap distance from A is larger, which better reflects the loss of route structure. This example demonstrates why edge overlap is more informative than traditional positional similarity for cyclic TSP routes. Figure 2 illustrates the concept. The attractiveness of a brighter firefly j to a dimmer firefly i is then defined as:

$$\beta_{ij} = \beta_0 \, exp\left(-\gamma \delta(\pi_i, \pi_j)\right) \tag{7}$$

where, β_0 is the base attractiveness and γ is the absorption coefficient. For each pair of pairs (π_i, π_j), the movement operator searched for edges that are in π_j but not in π_i and then tried to create one or more of these with a guided swap, insertion, or inversion move. The algorithm selected the operator to introduce the desired edge with the least impact on the quality of the subsequences. Each move kept permutations feasible, and only a small random perturbation was applied to avoid any collapse to the same attractor by the same operator.

3.3 Adaptive exploration, reheating, and embedded Two-Opt

Exploration was controlled by a monotonically decaying randomization schedule:

$$\alpha(t) = max\left(\alpha_{min}, \alpha_0 - (\alpha_0 - \alpha_{min})\frac{t}{T}\right) \tag{8}$$

where, α_0 and $\alpha_{\min}$ were the initial and minimum randomization strengths, t was the iteration index, and T was the total number of iterations. The scalar t in Eq. (8) is distinct from the travel-time function $t(i, j)$ in Eqs. (2)–(4). The parameter α_{reheat} denoted the reheating level, and k_{stag} denoted the stagnation threshold. This schedule supported broad exploration early in the search and more deterministic exploitation later. To prevent premature convergence, AHDFA monitored the improvement history of the global-best tour. If no improvement was recorded for k_{stag} consecutive iterations, $\alpha(t)$ was reheated to at least α_{reheat}, and the algorithm partially reinitialized a small fraction of the worst-performing fireflies. Diversification therefore increased only when stagnation was observed.

The implementation applied bounded Two-Opt to each modified tour after the guided movement phase. Two-Opt removed edge crossings by replacing two edges with two noncrossing edges when the exchange reduced total length [1, 2]. The bounded improvement budget prevented local search from dominating runtime while retaining a strong exploitation step. The resulting search was memetic: population interaction provided global route information and Two-Opt converted promising edge patterns into locally improved tours before the next attraction cycle. Figure 3 shows the workflow.

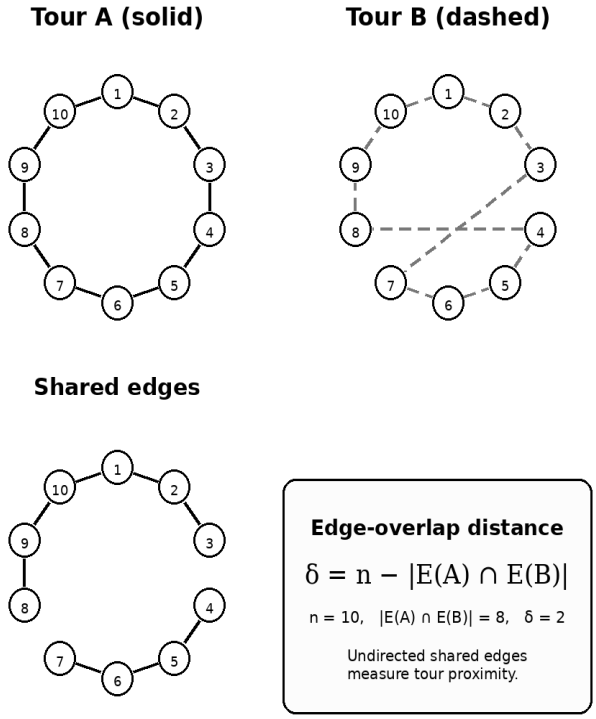

Figure 2. Edge-overlap distance concept used to compare two tours; shared undirected tour edges reduce the distance even when the same cities appear at shifted permutation positions

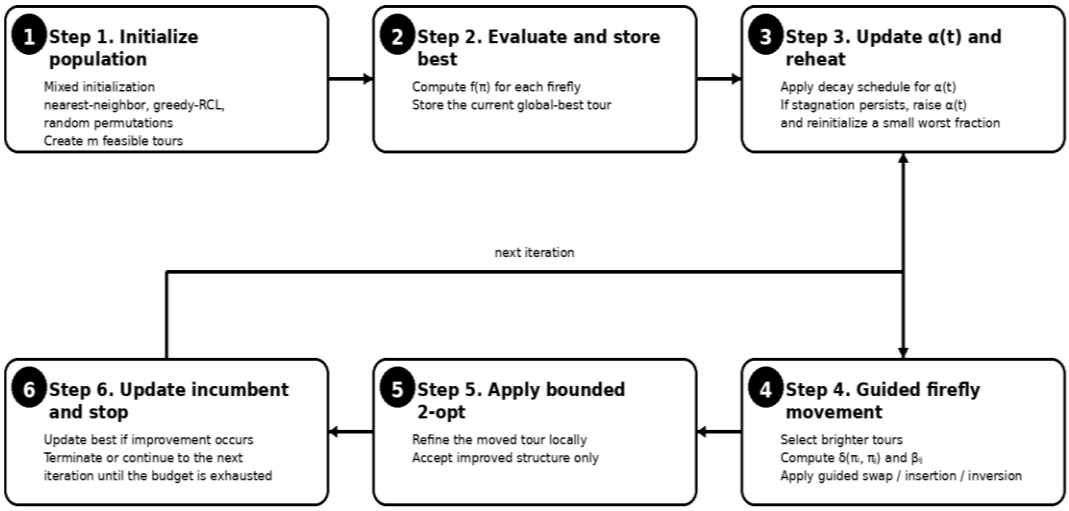

Figure 3. Workflow of the proposed Adaptive Hybrid Discrete Firefly Algorithm (AHDFA)

3.4 Computational complexity

Let m be the population size and q the maximum number of brighter fireflies considered by each solution in one iteration. Constructing route costs and edge sets takes $O(mn)$. Each attraction interaction uses $O(n)$ edge comparisons plus the cost of a bounded discrete edit sequence. If we denote the local search budget by BTwo-Opt, then the practical per-iteration cost becomes $O(mqn + mB\text{Two-Opt})$. In the worst case, a full Two-Opt neighborhood scan scales as $O(n^2)$, so local refinement is the module most likely to dominate asymptotic cost when the budget is not bounded.

The observed runtime pattern was consistent with this analysis. AHDFA averaged 5.73 s. It was slower than the computationally lighter Nearest Neighbor (NN) + Two-Opt (0.05 s), Simulated Annealing (SA) + Two-Opt (0.32 s), and Ant Colony System (ACS) + Two-Opt (2.16 s) comparators, but faster than Genetic Algorithm (GA) + Two-Opt (8.58 s) and substantially faster than Iterated Local Search (ILS) + Two-Opt (15.76 s). Repeated edge-set comparisons and bounded Two-Opt refinement accounted for most of the cost. Setting $q = 3$, limiting each interaction to one guided-move attempt, and using a Two-Opt budget of 6 prevented these modules from becoming full quadratic scans. Substantially larger routing instances would require stricter sampling, candidate-neighbor lists, parallel evaluation, or adaptive local-search budgets to maintain scalability.

4. EXPERIMENTAL DESIGN

4.1 Benchmarks and comparative algorithms

The benchmark study used 15 Euclidean TSPLIB instances spanning 51–150 cities: eil51, berlin52, st70, eil76, pr76, rat99, kroA100, eil101, lin105, pr124, bier127, ch130, pr136, pr144, and ch150 [25]. Table 1 lists their best-known tour lengths. The application study used a 10-scenario synthetic UAV-assisted IoT family rather than a single case. The scenarios varied in map dimensions (2.4 × 1.8 km to 3.8 × 3.0 km), depot placement (corner, center, or offset), cluster-head count (32–60), cruise speed (11–15 m/s), mean service time (14–26 s with jitter), and priority structure (uniform, banded, or gradient). The released scenarios are distance-based Euclidean routing inputs; obstacle and no-fly-zone constraints were not evaluated.

Table 1. Expanded Traveling Salesman Problem Library (TSPLIB) benchmark portfolio

Instance	Cities (n)	Best-Known Tour	Class
eil51	51	426	Small
berlin52	52	7542	Small
st70	70	675	Small
eil76	76	538	Medium
pr76	76	108159	Medium
rat99	99	1211	Medium
kroA100	100	21282	Medium
eil101	101	629	Medium
lin105	105	14379	Medium
pr124	124	59030	Medium
bier127	127	118282	Medium
ch130	130	6110	Medium
pr136	136	96772	Medium
pr144	144	58537	Medium
ch150	150	6528	Medium

AHDFA was compared with five comparator methods representing common routing and metaheuristic families: ACS + Two-Opt as a constructive swarm method; ILS + Two-Opt as a perturb-and-improve local-search method; GA + Two-Opt as a permutation-based evolutionary method; SA + Two-Opt as a single-solution Metropolis method; and NN + Two-Opt as a deterministic constructive reference. The GA implementation followed the classical formulations of Potvin [26] and Whitley [27], whereas the SA implementation followed Kirkpatrick et al. [28]. This comparator set covered population-based construction, evolutionary recombination, stochastic local search, and deterministic constructive routing under a common Euclidean closed-tour protocol. It was not intended to represent every specialized state-of-the-art TSP solver. Each stochastic method was run with 30 independent seeds (101–130). The benchmark analysis comprised 2,700 TSPLIB runs; the ablation and UAV-assisted IoT scenario studies contributed 900 and 1,800 additional runs, respectively.

Comparator transparency is supported by a common study protocol. All methods use the same symmetric Euclidean distance matrices, closed-tour representation and node sets. The added +Two-Opt sign indicates the bounded Two-Opt

post-improvement stage in the archived workflow after each base search stage or constructive initializer. AHDFA, ACS + Two-Opt, ILS + Two-Opt, GA + Two-Opt, and SA + Two-Opt were stochastic and used the common seed list; NN + Two-Opt was deterministic. The comparator families, seed policy, stopping rules, and provenance summary are preserved in data/comparator_protocol.csv. Because the companion archive is a manuscript-linked reproducibility bundle rather than the complete orchestration archive, the study provides a transparent common-protocol comparison rather than an exhaustive evaluation of fully tuned comparator methods.

4.2 Evaluation metrics and statistical protocol

For the TSPLIB instances, solution quality was evaluated using the relative percentage deviation from the best-known tour:

$$RPD(\pi) = 100 \times \frac{f(\pi) - f^*}{f^*} \tag{9}$$

where, f^* is the best-known tour length. Lower RPD values indicate better performance. Runtime was measured in seconds under the same implementation environment. Overall differences were assessed with Friedman's nonparametric rank test, and effect size was summarized with Kendall's W [29]. Pairwise follow-up comparisons used Wilcoxon signed-rank tests on instance-level mean RPD with Holm correction for multiplicity [29-31]. Rank-biserial correlations were reported as directional effect-size summaries. The analysis comprised overall benchmark summaries, Holm-corrected pairwise outcomes, ablation summaries, and a performance-profile figure based on instance-level mean RPD.

4.3 Reporting and reproducibility

To improve the transparency of the study, the companion package distinguishes the archived settings for the expanded study from the executable reference defaults. The official settings file, data/study_settings_broadened_study.json, provides the runtime-bounded configuration with a population size of $m = 7$, iteration budget of $T = 24$, $\beta_0 = 1.0$, $\gamma = 2.0$, $\alpha_0 = 0.30$, $\alpha_{min} = 0.05$, $\alpha_{reheat} = 0.20$ and $k_{stag} = 8$. It also specifies $q = 3$, one guided move, a Two-Opt budget of 6, reinitialization fraction of 0.15 and an edge sample size of 3. The executable reference configuration is stored in data/ahdfa_reference_default_config.json. The comparator families, seed policy, stopping rules and provenance summary are stored in data/comparator_protocol.csv, while docs/STUDY_EXECUTION_SCOPE.md explains how the records should be interpreted together. Parameter values were selected through conservative pilot calibration rather than exhaustive tuning. The compact population and iteration budgets were chosen to keep runtime comparable with the comparator methods. Moderate values of β_0 and γ allowed attraction to distinguish low- and high-overlap tours without causing immediate population collapse. The randomization parameter α decreased from exploratory to exploitative levels, and α_{reheat} was activated only after k_{stag} stagnant iterations. The limits $q = 3$, one guided move, and a Two-Opt budget of 6 balanced search quality against computational cost. The ablation analysis in Section 5.3 showed that reducing or removing Two-Opt degraded solution quality, whereas more intensive local search would increase runtime. Figure 4 presents the corresponding TSPLIB performance profile.

The companion package is organized as a manuscript-linked reproducibility bundle rather than a complete experiment-rerun archive. The reported settings are stored in data/study_settings_broadened_study.json, the executable defaults in data/ahdfa_reference_default_config.json, and the comparator stopping-rule and provenance information in data/comparator_protocol.csv. The file docs/STUDY_EXECUTION_SCOPE.md defines their scope. The package also contains the 15 TSPLIB instances, 10 UAV scenario files, machine-readable benchmark, ablation, and UAV summary tables, omnibus and Holm-corrected statistics, final figure files, scripts for regenerating packaged summaries and selected plots, environment descriptors, deposit metadata, and checksums. It therefore supports transparent inspection, report-level reproduction of the archived summary outputs, and unambiguous recovery of the study settings. However, it does not preserve every original orchestration file or raw run-level log.

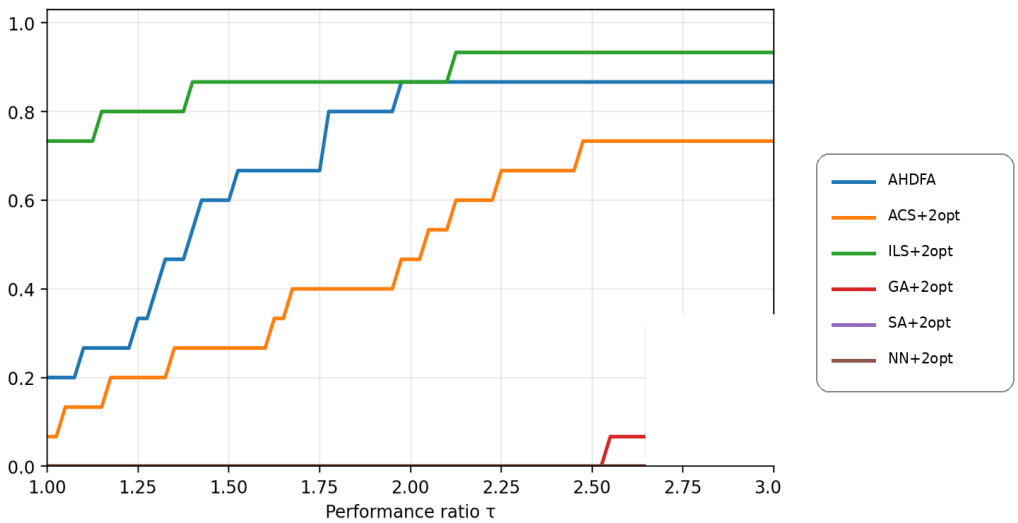

Figure 4. TSPLIB performance profile based on instance-level mean RPD: Curves closer to the upper-left indicate that a method reaches low-RPD thresholds on more instances; ILS + Two-Opt ranks first and AHDFA ranks second across the portfolio
Note: TSPLIB: Traveling Salesman Problem Library; RPD: Relative Percentage Deviation; ILS: Iterated Local Search; AHDFA: Adaptive Hybrid Discrete Firefly Algorithm.

5. RESULTS AND DISCUSSION

5.1 Traveling Salesman Problem Library solution quality

Table 2 summarizes the expanded TSPLIB comparison across the 15-instance portfolio. ILS + Two-Opt achieved the best overall average rank (1.33) and the lowest mean instance RPD (2.77%), while AHDFA ranked second (2.13) with a mean instance RPD of 3.40%. ACS + Two-Opt followed with 4.86%, whereas GA + Two-Opt, NN + Two-Opt, and SA + Two-Opt were substantially worse at 17.45%, 18.18%, and 62.18%, respectively. At the instance level, ILS + Two-Opt achieved the best mean RPD on 11 of the 15 problems, AHDFA on 3 (bier127, ch150, and lin105), and ACS + Two-Opt on 1 (pr144).

It is clear from Figure 4 that ILS + Two-Opt dominates the left side of the profile, while AHDFA is the second most competitive method among the larger instance set. Compared to the conventional comparators, AHDFA reduces mean instance RPD by almost 30.1% compared to ACS + Two-Opt, 80.5% compared to GA + Two-Opt, 81.3% compared to NN + Two-Opt, and 94.5% compared to SA + Two-Opt. The comparison with ILS + Two-Opt is a bit different: AHDFA was still competitive but did not achieve the lowest overall mean RPD.

Table 2. Overall Traveling Salesman Problem Library (TSPLIB) performance summary across 15 instances and 30 runs per stochastic method

Method	Avg Rank	Mean RPD	Median RPD	Mean Runtime (s)	Best Instance RPD	Worst Instance RPD
ILS + Two-Opt	1.33	2.77	2.52	15.76	1.07	6.84
AHDFA	2.13	3.40	2.78	5.73	1.90	6.06
ACS + Two-Opt	2.53	4.86	4.28	2.16	0.39	10.90
GA + Two-Opt	4.40	17.45	14.02	8.58	3.06	40.75
NN + Two-Opt	4.67	18.18	18.68	0.05	5.87	27.24
SA + Two-Opt	5.93	62.18	69.18	0.32	14.62	97.33

Note: RPD = Relative Percentage Deviation, ILS = Iterated Local Search, ACS = Ant Colony System, GA = Genetic Algorithm, NN = Nearest Neighbor, SA = Simulated Annealing.

5.2 Statistical and runtime analysis

The omnibus statistical analysis indicated that the ranking differences were consistent for the instances as opposed to being driven by one or two problems. For the quality of solutions, Friedman's test of instance-level mean RPD was 66.81 with $p = 4.72 \times 10^{-13}$ and Kendall's W = 0.891. This clearly indicates that the rank orders are very close for different instances. As shown in Table 3, our Holm-corrected pairwise comparisons showed that AHDFA is significantly better than GA + Two-Opt, NN + Two-Opt and SA + Two-Opt. However, the comparison with ACS + Two-Opt and ILS + Two-Opt were not significant at the 0.05 level, although among the two methods, ILS + Two-Opt had the lowest average RPD. This indicates that AHDFA is the best among these methods but does not demonstrate statistical superiority over all others. The basic strength of AHDFA compared with the others is its edge-aware interpretability, low RPD and moderate runtime, but not overall superiority. The runtime results showed a different trade-off. NN + Two-Opt was the fastest method in terms of time (0.05 s) and SA + Two-Opt (0.32 s) and ACS + Two-Opt (2.16 s) were the second most quickly. In the middle tier, 5.73 s was the fastest GA + Two-Opt (8.58 s) and ILS + Two-Opt (15.76 s) but slower than the easier constructive comparators and swarm comparators. Friedman's runtime analysis gave a value of 74.47 for the parameters $p = 1.20 \times 10^{-14}$ and Kendall's $W = 0.993$, indicating a near-perfect consistency of runtime order over the 15 instances.

5.3 Multi-instance ablation results

Table 4 shows the results of the multi-instance ablation for five common problems. Removing the embedded Two-Opt stage significantly worsened the results. The mean instance RPD for the full method increased from 2.89% to 18.48% without the embedded Two-Opt. This result has pointed to the bounded local refinement as being the most important component in terms of the current search budget. The interaction between edge-overlap attraction and Two-Opt can be seen to be a key factor. Edge overlap leads the population to tours which have some useful adjacencies, while Two-Opt transforms the adjacency patterns into shorter noncrossing segments. The removal of edge-overlap guidance led to mean instance RPD increasing to 3.15%, and the reduction of the Two-Opt budget increased it to 4.06%, indicating that the global edge signal and the local edge repair were interdependent. The reheating effect was more sensitive to the computational budget, with the "No reheating" version slightly better than the full setup in this five-instance ablation (2.85% vs 2.89%). Overall, the ablation showed that embedded Two-Opt was needed, edge-aware attraction was beneficial, and diversification should be appropriate for the available budget and problem set.

Table 3. Holm-corrected pairwise post-hoc comparison of AHDFA against comparator methods on instance-level mean RPD

Comparator	Mean RPD	Median Diff	Holm p	RBC	Result
GA + Two-Opt	17.45	–11.60	0.0009	–1.00	AHDFA better
NN + Two-Opt	18.18	–14.63	0.0009	–1.00	AHDFA better
SA + Two-Opt	62.18	–64.69	0.0009	–1.00	AHDFA better
ACS + Two-Opt	4.86	–1.36	0.1060	–0.62	Not significant
ILS + Two-Opt	2.77	0.82	0.1060	0.58	Not significant

Note: RPD = Relative Percentage Deviation, AHDFA = Adaptive Hybrid Discrete Firefly Algorithm, ILS = Iterated Local Search, ACS = Ant Colony System, GA = Genetic Algorithm, NN = Nearest Neighbor,SA = Simulated Annealing, RBC = rank-biserial correlation.

Table 4. Multi-instance ablation summary

Variant	Mean RPD	Runtime (s)
No reheating	2.85	5.48
Full AHDFA	2.89	5.35
Swap only	2.95	5.33
No edge overlap	3.15	2.17
Reduced Two-Opt	4.06	1.79
No embedded Two-Opt	18.48	0.32

Note: RPD = Relative Percentage Deviation, AHDFA = Adaptive Hybrid Discrete Firefly Algorithm.

5.4 UAV scenario-family study

Table 5 summarizes the 10-scenario UAV family. Compared with the original single-case study, the scenario-family evaluation produced a more nuanced ranking. ILS + Two-Opt achieved the best average distance rank (1.3), ACS + Two-Opt ranked second (2.2), and AHDFA ranked third (2.5). AHDFA averaged 14,068.75 m across the scenario family, compared with 13,938.70 m for ILS + Two-Opt and 14,053.55 m for ACS + Two-Opt. The average gap between AHDFA and ILS + Two-Opt was only 130.05 m (0.93%), and the gap between AHDFA and ACS + Two-Opt was 15.20 m (0.11%).

Although AHDFA did not achieve the best mean distance in the UAV study, it remained competitive and outperformed GA + Two-Opt, SA + Two-Opt, and NN + Two-Opt. Relative to these methods, AHDFA reduced mean route length by approximately 1.99%, 7.16%, and 11.03%, respectively. The weighted-arrival proxy showed a similar pattern: AHDFA achieved the second-lowest mean value (62,275.1), close to ILS + Two-Opt (62,262.5) and slightly lower than ACS + Two-Opt (62,331.7). These results show that the structure-aware search remained competitive in both route length and freshness, even though it did not achieve the lowest mean route distance. Figure 5 presents the mean route-distance comparison for the scenario family.

Table 5. Overall UAV scenario-family summary across 10 synthetic routing scenarios

Algorithm	Average Distance Rank	Mean Distance (m)	Mean Weighted-Arrival Proxy
ILS + Two-Opt	1.3	13938.70	62262.5
ACS + Two-Opt	2.2	14053.55	62331.7
AHDFA	2.5	14068.75	62275.1
GA + Two-Opt	4.0	14354.56	63145.2
SA + Two-Opt	5.1	15153.62	65404.2
NN + Two-Opt	5.9	15812.69	66333.3

Note: AHDFA = Adaptive Hybrid Discrete Firefly Algorithm, ILS = Iterated Local Search, ACS = Ant Colony System, GA = Genetic Algorithm, NN = Nearest Neighbor, SA = Simulated Annealing.

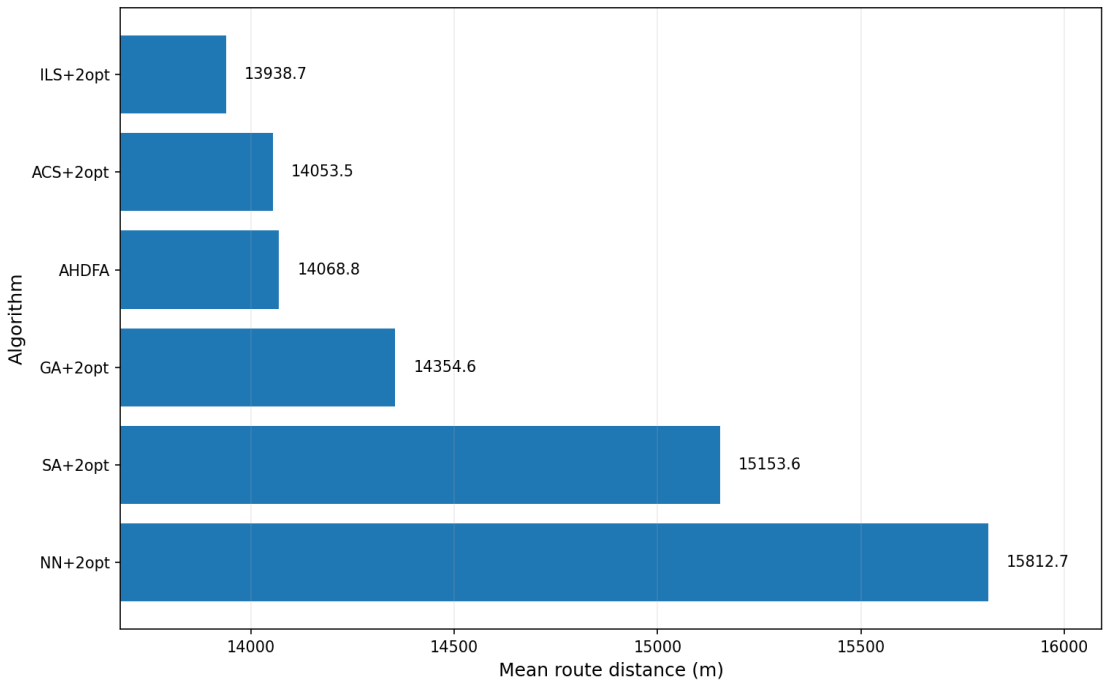

Figure 5. Mean route distance across the 10-scenario UAV family

Note: Lower values are better. AHDFA remained close to ACS + Two-Opt and ILS + Two-Opt while outperforming GA + Two-Opt, SA + Two-Opt, and NN + Two-Opt. UAV = unmanned aerial vehicle; AHDFA = Adaptive Hybrid Discrete Firefly Algorithm; ILS = Iterated Local Search; ACS = Ant Colony System; GA = Genetic Algorithm; NN = Nearest Neighbor; SA = Simulated Annealing.

5.5 Discussion and limitations

The results from the expanded experiments lead to a limited conclusion: AHDFA is a competitive, structure-aware routing algorithm with a functional mechanism for interpretation but it is not a universally superior solver. The performance remained similar across the wider TSPLIB toolkit; the ablation results justified the embedded local search analysis; and the UAV scenarios were close to the best distance-focused methods and also showed good freshness performance. The

expanded approach to the experiments provided a more thorough understanding of the situation, but there are still some limitations. The UAV scenarios were synthetic and not field-derived. Future work should be based on publicly available UAV traces or field-based flight data or tested digital-twin scenarios. In practice, obstacle avoidance and no-fly-zone constraints, wind and battery issues, limited communication periods for data download, and multi-UAV collaboration are all important. This may affect the route-cost function and the possibility of edge accessibility. The current edge-overlap operator must be combined with constraint-aware edge filtering or repair. The benchmark portfolio is larger than the pilot study, but it did not include larger examples and specialized route-improvement methods such as Lin-Kernighan and EAX solvers. We excluded these methods as we used a relatively small set of comparators under a single archived workflow, rather than a comprehensive comparison of state-of-the-art solvers. Also, we used a small set of comparisons, so some design choices, such as reheating, might not work well with a larger budget.

6. CONCLUSIONS

In this paper, we developed AHDFA for TSP-based UAV-assisted IoT route planning. We defined attraction as edge overlap, transformed attraction into feasible guided permutation edits, reheated exploration after stagnation and applied bounded Two-Opt refinement. The resulting model combines interpretable global search with controlled local exploitation. In the extended TSPLIB portfolio of methods, we found that AHDFA was the best and significantly outperformed the other local search-based methods. However, our results were not significant for the comparison with the local search-based comparators. We found that embedded Two-Opt is the key driver of our approach, and we confirmed the role of edge-overlap guidance. In the synthetic UAV scenarios, AHDFA remained competitive in route length and had good freshness-related performance. These results indicate that AHDFA is a competitive, structure-aware routing method instead of a universal best solver. In the future, we will evaluate larger benchmark suites and field-informed UAV datasets and include communication and energy constraints, obstacle and no-fly-zone modeling, and multi-UAV coordination.

DATA AND CODE AVAILABILITY

The companion reproducibility package is available on Zenodo at https://doi.org/10.5281/zenodo.19648122. It preserves the study settings, comparator-protocol records, scenarios, tables, figures, and checksums needed to inspect the expanded study and reproduce the packaged summary outputs. The deposit is a companion reproducibility bundle rather than a complete experiment-rerun archive.

AUTHOR CONTRIBUTIONS

Meaad Mohammed Salih: Conceptualization, methodology, software, writing – original draft, and visualization. Raya Basil Alothman: Formal analysis, validation, and writing – review and editing. Ali A. Al-Arbo: Supervision, investigation, resources, and writing – review and editing.

REFERENCES

[1] Croes, G.A. (1958). A method for solving traveling-salesman problems. Operations Research, 6(6): 791-812. https://doi.org/10.1287/opre.6.6.791

[2] Lin, S., Kernighan, B.W. (1973). An effective heuristic algorithm for the traveling-salesman problem. Operations Research, 21(2): 498-516. https://doi.org/10.1287/opre.21.2.498

[3] Messaoudi, K., Oubbati, O.S., Rachedi, A., Lakas, A., Bendouma, T., Chaib, N. (2023). A survey of UAV-based data collection: Challenges, solutions and future perspectives. Journal of Network and Computer Applications, 216: 103670. https://doi.org/10.1016/j.jnca.2023.103670

[4] Amodu, O.A., Bukar, U.A., Raja Mahmood, R.A., Jarray, C., Othman, M. (2023). Age of information minimization in UAV-aided data collection for WSN and IoT applications: A systematic review. Journal of Network and Computer Applications, 216: 103652. https://doi.org/10.1016/j.jnca.2023.103652

[5] Cheng, N., Wu, S., Wang, X.C., et al. (2023). AI for UAV-assisted IoT applications: A comprehensive review. IEEE Internet of Things Journal, 10(16): 14438-14461. https://doi.org/10.1109/JIOT.2023.3268316

[6] Wang, S.Q., Qi, N., Jiang, H., et al. (2024). Trajectory planning for UAV-assisted data collection in IoT network: A double deep Q network approach. Electronics, 13(8): 1592. https://doi.org/10.3390/electronics13081592

[7] Yang, X.S., He, X.S. (2013). Firefly algorithm: Recent advances and applications. International Journal of Swarm Intelligence, 1(1): 36-50. https://doi.org/10.1504/IJSI.2013.055801

[8] Yang, X.S. (2010). Firefly algorithm, stochastic test functions and design optimisation. International Journal of Bio-Inspired Computation, 2(2): 78-84. https://doi.org/10.1504/IJBIC.2010.032124

[9] Fister, I., Fister Jr., I., Yang, X.S., Brest, J. (2013). A comprehensive review of firefly algorithms. Swarm and Evolutionary Computation, 13: 34-46. https://doi.org/10.1016/j.swevo.2013.06.001

[10] Kumar, V., Kumar, D. (2021). A systematic review on firefly algorithm: Past, present, and future. Archives of Computational Methods in Engineering, 28: 3269-3291. https://doi.org/10.1007/s11831-020-09498-y

[11] Li, J., Wei, X.Y., Li, B., Zeng, Z.G. (2022). A survey on firefly algorithms. Neurocomputing, 500: 662-678. https://doi.org/10.1016/j.neucom.2022.05.100

[12] Zhou, L., Ding, L., Qiang, X., Luo, Y. (2015). An improved discrete firefly algorithm for the traveling salesman problem. Journal of Computational and Theoretical Nanoscience, 12(7): 1184-1189. https://doi.org/10.1166/jctn.2015.3871

[13] Teng, L., Li, H. (2018). Modified discrete firefly algorithm combining genetic algorithm for traveling salesman problem. TELKOMNIKA (Telecommunication Computing Electronics and Control), 16(1): 424-431. https://doi.org/10.12928/telkomnika.v16i1.4752

[14] Alorf, A. (2023). A survey of recently developed

metaheuristics and their comparative analysis. Engineering Applications of Artificial Intelligence, 117: 105622. https://doi.org/10.1016/j.engappai.2022.105622
[15] Toaza, B., Esztergár-Kiss, D. (2023). A review of metaheuristic algorithms for solving TSP-based scheduling optimization problems. Applied Soft Computing, 148: 110908. https://doi.org/10.1016/j.asoc.2023.110908
[16] Kosta, A., Pappas, N., Angelakis, V. (2017). Age of information: A new concept, metric, and tool. Foundations and Trends in Networking, 12(3): 162-259. https://doi.org/10.1561/1300000060
[17] Sun, Y., Uysal-Biyikoglu, E., Yates, R.D., Koksal, C.E., Shroff, N.B. (2017). Update or wait: How to keep your data fresh. IEEE Transactions on Information Theory, 63(11): 7492-7508. https://doi.org/10.1109/TIT.2017.2735804
[18] Yates, R.D., Sun, Y., Brown III, D.R., et al. (2021). Age of information: An introduction and survey. IEEE Journal on Selected Areas in Communications, 39(5): 1183-1210. https://doi.org/10.1109/JSAC.2021.3065072
[19] Wang, T.L., Fu, X.W. Guerrieri, A. (2024). Joint resource scheduling and flight path planning of UAV-assisted IoTs in response to emergencies. Computer Networks, 253: 110731. https://doi.org/10.1016/j.comnet.2024.110731
[20] Alqefari, S., Menai, M.E.B. (2025). Multi-UAV task assignment in dynamic environments: Current trends and future directions. Drones, 9(1): 75. https://doi.org/10.3390/drones9010075
[21] Satouf, A., Hamidoğlu, A., Gul, O.M., Kuusik, A., Kadry, S.N., Elghirani, A. (2025). A survey on task scheduling and optimization techniques for IoT-enabled UAV with edge/fog computing. Telecommunication Systems, 88(3): 89. https://doi.org/10.1007/s11235-025-01320-z
[22] Al-Arbo, A.A., Al-Arbo, Y., Salih, M.M. (2026). Swarm robotics: Optimizing collective behavior through advanced metaheuristic algorithms. Journal Européen des Systèmes Automatisés, 59(2): 559-565. https://doi.org/10.18280/jesa.590225
[23] Al-Arbo, A.A., Al-Arbo, Y. (2026). Adaptive random-key particle swarm optimization with DC-closure local search for a two-stage fixed-charge transportation benchmark. International Journal of Intelligent Engineering and Systems, 19(4): 899-911. https://doi.org/10.22266/ijies2026.0430.51
[24] Al-Arbo, A.A., Al-Kawaz, R.Z., Jameel, M.S. (2025). New meta-heuristic computer-oriented algorithms to solve unconstrained optimization problems. Mathematical Modelling of Engineering Problems, 12(3): 1081-1089. https://doi.org/10.18280/mmep.120335
[25] Reinelt, G. (1991). TSPLIB--A traveling salesman problem library. ORSA Journal on Computing, 3(4): 376-384. https://doi.org/10.1287/ijoc.3.4.376
[26] Potvin, J.Y. (1996). Genetic algorithms for the traveling salesman problem. Annals of Operations Research, 63: 337-370. https://doi.org/10.1007/BF02125403
[27] Whitley, D. (1994). A genetic algorithm tutorial. Statistics and Computing, 4: 65-85. https://doi.org/10.1007/BF00175354
[28] Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P. (1983). Optimization by simulated annealing. Science, 220(4598): 671-680. https://doi.org/10.1126/science.220.4598.671
[29] Derrac, J., García, S., Molina, D., Herrera, F. (2011). A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. Swarm and Evolutionary Computation, 1(1): 3-18. https://doi.org/10.1016/j.swevo.2011.02.002
[30] Holm, S. (1979). A simple sequentially rejective multiple test procedure. Scandinavian Journal of Statistics, 6(2): 65-70. http://www.jstor.org/stable/4615733.
[31] García, S., Fernández, A., Luengo, J., Herrera, F. (2010). Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power. Information Sciences, 180(10): 2044-2064. https://doi.org/10.1016/j.ins.2009.12.010

NOMENCLATURE

$A\pi(k)$	arrival time at the k-th visited node
$d(i,j)$	symmetric travel cost or distance between nodes i and j
$E(\pi)$	set of undirected tour edges induced by route π
$f(\pi)$	distance-only route objective
$g(\pi)$	weighted freshness proxy based on arrival times
$I(\pi)$	firefly brightness used for minimization
$s(i)$	service or hovering time at node i
$t(i,j)$	travel time between nodes i and j
V	set of nodes including depot and cluster heads
0	depot node from which the UAV departs and to which it returns

Greek symbols

$\alpha(t)$	randomization strength at iteration t
β_{ij}	attractiveness of firefly j to firefly i
$\delta(\pi_i,\pi_j)$	edge-overlap distance between tours π_i and π_j
π	permutation describing the visit order of cluster heads

Subscripts

i,j	indices of compared fireflies or tours
k,h	visit-order indices in a route

APPENDIX

Appendix A. Archived comparator settings used in the common-protocol study

Table A1 summarizes the archived comparator settings used in the common-protocol study.

Table A1. Archived comparator settings used in the common-protocol study

Item	Archived Setting or Interpretation
AHDFA	Study settings: $m = 7$; $T = 24$; $\beta_0 = 1.0$; $\gamma = 2.0$; $\alpha_0 = 0.30$; $\alpha_{min} = 0.05$; $\alpha_{reheat} = 0.20$; $k_{stag} = 8$; $q = 3$; a maximum of one

	guided move; Two-Opt budget = 6; reinitialization fraction = 0.15; edge sample size = 3; stochastic seeds 101–130.
Stochastic comparator baselines	ACS + Two-Opt, ILS + Two-Opt, GA + Two-Opt, and SA + Two-Opt used the same Euclidean closed-tour representation, node sets, cost model, stochastic seed range, and bounded Two-Opt post-improvement. The manuscript-linked archive does not fully preserve the low-level, family-specific controls.
Deterministic baseline	NN + Two-Opt used the same node sets, representation, cost model, and bounded Two-Opt post-improvement, without random seeds.
Archive scope	The appendix and companion records document the common protocol and the limits of interpretation; they do not claim to preserve every original orchestration file or every raw per-run log.