A Hybrid Local Search and Genetic Algorithm with Enhanced Population Initialization for the Traveling Salesman Problem

Wannaporn Teekeng[†], Non-member

ABSTRACT

The Traveling Salesman Problem (TSP) remains a fundamental NP-hard combinatorial optimization challenge with significant theoretical and practical implications. While Genetic Algorithms (GAs) have demonstrated considerable promise for solving TSP instances, their performance is often hindered by poor initial population quality, leading to slow convergence and suboptimal solutions, particularly for large-scale problems. This paper proposes a hybrid initialization strategy for Genetic Algorithms applied to TSP, implemented within the HLGA framework. The main contribution combines k-opt refined solutions (2-opt and 3-opt) with random permutations to create an enriched initial population. This strategy accelerates convergence while maintaining diversity and solution space exploration capabilities. Comprehensive experiments conducted on 27 TSPLIB benchmark instances (30-200 cities) demonstrate HLGA's superior performance compared to standalone 3-opt heuristics and five state-of-the-art algorithms (GGSC-SSA, SCGA, IMODE/J2020, ACO-DSA, and RL-SA). Notably, HLGA achieves or surpasses the Best-Known Solution (BKS) for 11 instances, with an overall average relative error of 9.22%—significantly outperforming the 3-opt baseline (83.91%). These results validate that integrating targeted local search into population initialization substantially enhances both convergence speed and solution quality, establishing HLGA as a competitive and scalable approach for large-scale TSP optimization

Keywords: Traveling Salesman Problem, Genetic Algorithm, Local Search, K-opt

1. INTRODUCTION

The Traveling Salesman Problem (TSP) and its generalized form, the Generalized Traveling Salesman Problem (GTSP), are fundamental and extensively studied combinatorial optimization problems in computer science. The TSP is categorized as an NP-hard problem, implying that no known algorithm can solve all instances efficiently within polynomial time. As the number of cities increases, the complexity of determining the optimal solution escalates significantly. TSP variants are classified into two primary categories: symmetric and asymmetric[1-2]. In the Symmetric Traveling Salesman Problem (STSP), the distance between any two cities is identical in both directions, formally defined as $d(i,j) = d(j,i)$ for all pairs of cities $i,j \in V$, where V denotes the set of cities. In contrast, the Asymmetric Traveling Salesman Problem (ATSP) allows for directional differences in distances, such that $d(i,j) \neq d(j,i)$.

Recent research has advanced algorithm development to achieve faster convergence by refining local search techniques to avoid local optima. GGSC-SSA model used Greedy Algorithm, Genetic Operators, an adaptive weighting mechanism, and a Sine-Cosine Search Strategy [3]. SCGA model, a hybridization of the Genetic Algorithm (GA) and Simulated Annealing (SA), incorporates an elitist strategy to accelerate convergence [4]. A novel dataset was constructed from real-world data of 20 cities. Among the evaluated algorithms, IMODE and j2020 demonstrated superior performance [5]. ACO-DSA algorithm, a hybrid approach integrating Ant Colony Optimization (ACO) and Simulated Annealing (SA), enhances route optimization. It initially employs a clustering phase to partition routes into sub-groups. Subsequently, these sub-groups are iteratively refined by the algorithm, resulting in routes with significantly improved efficiency [6], Hyper-heuristic approach employ ensembles of heuristics, specifically priority rules generated via Genetic Programming (GP) [7]. RL-SA model integrates Deep Reinforcement Learning (DRL) with Simulated Annealing (SA), the Whale Optimization Algorithm (WOA), and Gaussian Perturbation. The model is designed to autonomously learn path improvement strategies, thereby obviating the need for complex, manually-defined rules [8]. The development from this diverse set of algorithms has been applied to other problems, such as reducing challenges in high-speed wireless communication, where noise often diminishes the data performance of end users. Therefore, ongoing efforts focus on devising robust algorithms that can achieve globally optimal solutions under various operational scenarios [9]. Meta-heuristic algorithms (PSO, SCA, WOA) tune Proportional-Integral (PI) controller parameters. The objective is to identify Kp and Ki values that ensure optimal control system stability [10]

Manuscript received on September 11, 2025; revised on October 29, 2025; accepted on January 6, 2026. This paper was recommended by Associate Editor Siraporn Sakphrom.

The author is with Rajamangala University of Technology Lanna, Chiang Mai, Thailand.

[†]Corresponding author: wannaporn@rmutl.ac.th

Digital Object Identifier: 10.37936/ecti-eec.2026241.262111

and modified genetic algorithm for flexible job-shop Scheduling problems [11].

A variety of algorithmic approaches have been developed to address TSP instances, metaheuristic and approximation algorithms iteratively refine solutions from an initial state, balancing solution quality and computational resources effectively. These methods have gained significant popularity due to their flexibility and efficiency in generating near-optimal solutions within reasonable time constraints. Achieving near-optimal results for the Traveling Salesman Problem (TSP) critically depends on generating high-quality initial solutions, in conjunction with employing effective main solution algorithms. Advanced learning techniques, such as deep learning in the TSPNet model and reinforcement learning in Neural-3-OPT [12-13], are applied for this objective. Additionally, constructive or insertion strategies based on mathematical principles, including the Girding Polygon heuristic and the Half Max Insertion Heuristic, offer another avenue for developing initial solutions, which can then be further refined using traditional local search techniques [14-15].

Recent research has extensively employed diverse local search strategies to enhance solution-finding processes, highlighting the crucial role of high-quality initial solutions in guiding search directions, improving solution quality, and accelerating convergence towards optimal outcomes. To address this issue, this paper introduces a Hybrid Local Search Genetic Algorithm (HLGA), integrating the k-opt local search technique to generate superior initial populations. This approach aims to overcome the limitations of traditional methods such as the Nearest Neighbour algorithm, which, despite its rapid execution, often exhibits poor performance and accuracy when solving large-scale problems. The proposed HLGA enhances initial populations before the optimization phase executed by Genetic Algorithms (GA), focusing on delivering high-quality solutions capable of effectively approaching the optimal solution of the Traveling Salesman Problem (TSP).

2. METHODOLOGY

2.1 Traveling Salesman Problem

Traveling Salesman Problem (TSP) is a well-known combinatorial optimization problem in computer science, involving the determination of an optimal route for traversing a set of locations. In the Traveling Salesman Problem (TSP), the input is an undirected graph $G = (V,E)$, where each edge $e \in E$ has an associated positive cost $c(e) > 0$. The objective is to determine a Hamiltonian cycle—a cycle that visits each vertex in V exactly once—that minimizes the total traversal cost.[5] Formally, given a set of N distinct cities, the goal is to find the shortest possible path that visits each city exactly once and returns to the original city. For instance, given a set of cities $N = \{1,2,3,4,5\}$. The tour follows the sequence: 1→2→3→4 →5→1, ensuring each city is visited exactly once before returning to the starting point.

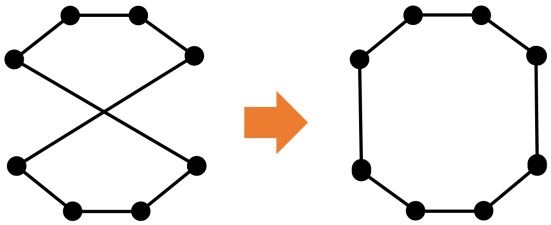

***Fig. 1**: 2-opt.*

Let D_{ij} denote the distance between node i and node j, and x_{ij} be a path from node i and node j

$$x_{ij} = \begin{cases} 1, & \text{if the tour from node } i \text{ to node } j \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

2.2 K-opt

The K-opt technique, a local search heuristic initially proposed by Croes (1958) [17] and subsequently refined by Flood, iteratively enhances solutions for the Traveling Salesman Problem (TSP). The core principle involves removing K edges from an existing tour and then reconnecting the resulting K path segments to form a new, potentially shorter, valid tour. While K-opt explores solution neighborhoods to improve tour quality, it has inherent limitations: convergence to local optima rather than the global optimum is possible, and its computational complexity grows exponentially with K, significantly increasing solution time. Fig. 1 conceptually illustrates a 2-opt move [18].

The iterative improvement procedure using 2-opt comprises the following steps:

1) Edge Selection: Identify two non-adjacent edges in the current tour, for instance, (v_i, v_{i+1}) and (v_j, v_{j+1})
2) 2-opt Swap: Replace these selected edges with two new edges, (v_i, v_j) and (v_{i+1}, v_{j+1}). This is achieved by reversing the order of nodes in the tour segment strictly between v_i and v_{j+1} (or v_{i+1}) and v_j, depending on the chosen pair for reconnection).
3) Evaluation and Update: If this modification results in a shorter total tour length, the new tour is adopted.

To improve the quality of the initial solution, local search heuristics such as 2-opt and 3-opt are incorporated. The 2-opt technique systematically removes two non-adjacent edges and reverses the intervening segment, thereby eliminating potential crossovers and reducing the tour length. The 3-opt heuristic extends this idea by selecting three edges and evaluating various reconnection strategies, including segment reordering and selective reversals, to explore a broader neighborhood of the solution space.

These local search procedures are applied iteratively. The algorithm continues to perform successive 2-opt or 3-opt moves as long as they result in an improvement. Once

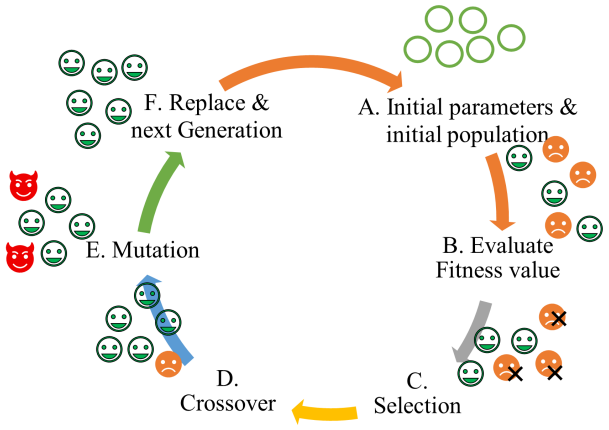

***Fig. 2**: An Operation of the GA.*

no further edge exchange leads to a reduction in tour length, the search terminates. At this stage, the current solution is considered a locally optimal tour, indicating that no further improvements are attainable within the explored neighborhood structure.

2.3 Genetic Algorithm

Genetic Algorithms (GAs) (Holland; Goldberg, 1989) [19] are metaheuristics emulating natural selection. The algorithm begins with an initial population of chromosomes, each encoding a candidate solution. Once the population is initialized, a fitness function evaluates each chromosome. Genetic operators—namely selection, crossover, and mutation—are then applied to produce a new generation of chromosomes. Those chromosomes in the current population whose fitness values fall below those of the newly generated offspring are replaced. This evolutionary cycle is iterated until either a predefined number of generations is reached or the fitness of the best chromosome satisfies a specified convergence criterion [20].

Genetic Algorithms (GA) are population-based metaheuristics inspired by natural selection, widely used for solving complex optimization problems. Their operation involves the following key steps:

a. Initialization: An initial population of chromosomes is generated, and a fitness function is defined to quantify solution quality.
b. Evaluation: The fitness of each chromosome in the current population is computed using the fitness function.
c. Selection: Parent chromosomes are probabilistically chosen based on their fitness (e.g., via roulette wheel or tournament selection), favoring higher-quality solutions.
d. Crossover: Selected parents are recombined with a crossover probability (pc, typically pc $\in$ [0.6,0.9]) to produce offspring, aiming to exploit beneficial solution structures.
e. Mutation: Offspring undergo random modifications with a low mutation probability (pm) to introduce genetic diversity and prevent premature convergence.
f. Replacement: A new population for the subsequent generation is formed from parents and offspring, often incorporating elitism (preserving the fittest individuals).

This evolutionary cycle repeats until a predefined termination criterion is met, yielding an optimal or near-optimal solution. Both GA and K-opt aim to find the optimal solution; their operational characteristics, advantages, and disadvantages can be compared as shown in Table 1.

2.4 Related Work on Population Initialization Strategies

The generation of a high-quality and diverse initial population is a critical factor influencing the performance of population-based algorithms, particularly for NP-hard optimization problems such as the Traveling Salesman Problem (TSP). Prior research has mainly explored two directions for improving initial population construction.

The first direction focuses on structured or patterned initialization designed to enhance early solution quality. Lin and Li proposed the Path-Guided Initialization (PGI) for the Capacitated TSP, where short path segments are constructed from nearest-neighbor candidate sets before inserting remaining cities, accelerating convergence and improving initial solution quality [21]. Similarly, Ibada et al. introduced the Circle Group Heuristic (CGH), a spatially structured initialization strategy that arranges initial tours to increase dispersion in DBMEA, resulting in higher-quality starting solutions and reduced computation time [22].

The second direction employs random or semi-random initialization followed by immediate local refinement. Wang et al. adopted this approach by generating random tours and improving all individuals using 2-opt local search to strengthen the baseline population before memetic evolution [23]. Likewise, Al-Ibrahim et al. used standard GA initialization combined with early-stage local search (NLSA) and a novel crossover operator to accelerate convergence beyond that of a traditional GA [24]. In contrast, Singh's group-theory-based initialization emphasizes positional diversity by assigning each chromosome a unique starting city (when population size equals the number of cities) and applies refinement through diversity-preserving crossover and 2-opt mutation [25].

Taken together, prior studies demonstrate that investing in initial solution quality or early refinement has a substantial impact on convergence speed and final solution performance. However, existing approaches generally emphasize either quality (e.g., PGI, CGH, local refinements) or diversity (e.g., group-theory initialization), with few methods integrating both in a unified manner.

To address this gap, the proposed HLGA employs a hybrid initialization strategy that generates approximately 10% locally refined solutions via 2-opt and 3-

***Table 1**: Comparation between GA and K-opt LS.*

Model	Genetic Algorithm (GA)	K-opt Local Search
Characteristics	Inspired by natural evolution such as selection, crossover, and mutation to discover optimal solutions in complex search spaces	Iteratively improves a given tour by swapping three edges to create a shorter route.
Advantages	- Effective for large-scale problems. - Flexible parameter tuning. - Learns from evolutionary feedback.	- Simple to implement and understand. - Enhances the quality of solutions, especially those generated by GA.
Disadvantages	- High computational cost for large instances. - Requires careful parameter tuning.	- May get stuck in local optima. - Performance depends on the initial solution.

opt heuristics, together with 90% randomly permuted tours. This balanced design introduces high-quality elite seeds while maintaining strong population diversity. As demonstrated in Section 4, this initialization significantly enhances GA search effectiveness, reduces premature convergence, and enables HLGA to outperform 3-opt and achieve objective values lower than the reported TSPLIB BKS in several datasets.

3. HLGA MODEL

The Traveling Salesman Problem (TSP) continues to be a pivotal NP-hard combinatorial optimization challenge due to its theoretical complexity and broad practical applicability. While Genetic Algorithm based approaches, including models like the Hybrid Local Search and Genetic Algorithm (HLGA), have been developed for constrained TSP variants, their efficacy often diminishes for large-scale instances. This is primarily attributed to extended computational times stemming from suboptimal initial solutions that impede convergence. To mitigate this, the proposed HLGA framework incorporates multiple heuristic techniques to augment both the diversity and quality of the initial population. Specifically, initial solutions are generated from two distinct, complementary strategies:

1) Local Refinement: K-opt heuristics (2-opt and 3-opt local search methods) are employed. These are selected for their established balance between yielding high-quality solutions and maintaining computational efficiency.
2) Initial population diversification: Randomized permutation-based initialization, leveraging principles from traditional GA operators, is utilized to increase population heterogeneity.

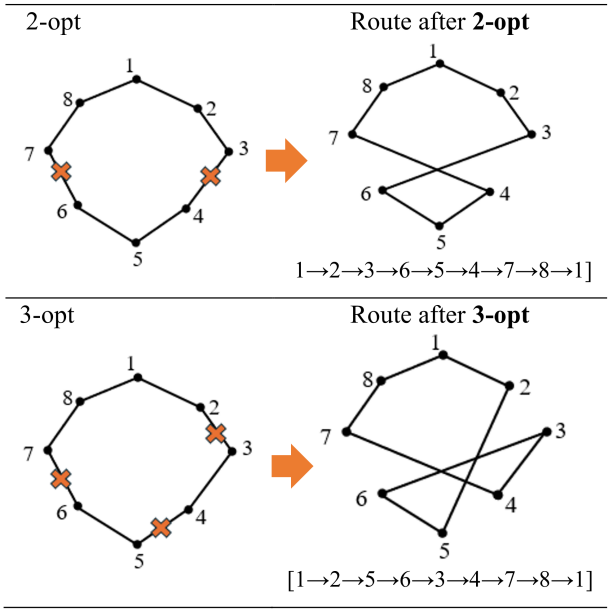

***Fig. 3**: Initial solutions with local search.*

By integrating targeted heuristic refinement with stochastic diversity in the initial population, this HLGA framework aims to accelerate convergence and improve final solution quality for large-scale TSP instances.

A permutation encoding scheme is employed to represent potential solutions, where each chromosome corresponds to a specific sequence of cities. For instance, a route consisting of 8 cities is encoded as [1, 2, 3, 4, 5, 6, 7, 8]. The decoding process involves calculating the total tour length by interpreting the permutation as a round-trip route that returns to the starting city. As a result, the complete tour is represented as [1, 2, 3, 4, 5, 6, 7, 8, 1], as illustrated in Fig. 3.

The initial solutions were generated using local search strategies, particularly employing 2-opt and 3-opt heuristics, which are designed to enhance the quality of partial tours within the overall route. These heuristics iteratively eliminate inefficient segments by performing edge exchanges that reduce the total path length. The local search process terminates when no further improvements are found, which is determined when the best-found solution remains unchanged for a number of consecutive iterations equal to the total number of cities (*n_points*) in the instance. To balance quality and computation time, the number of these locally optimized solutions (N) is limited to 10% of the total *n_points*. High-quality initial solutions obtained from local search are integrated with a population generated using a permutation-based approach for the Genetic Algorithm (GA). For example, in the case of the *eil51*

instance (51 cities) with a GA population size of 200, five tours may be derived through local search (10% of the total number of cities), while the remaining individuals are generated via random permutations. A subset of the top 155 tours, based on fitness evaluation, is then selected to construct the initial population. This hybrid initialization strategy combines locally optimized solutions with diverse permutation-based individuals, thereby enhancing population diversity and providing a more effective starting point for the GA search process.

During the solution search process involving tour reordering, the crossover operation employs the Order Crossover (OX) technique. This method selects a segment of the tour from one parent and fills the remaining positions with cities from the second parent, preserving their relative order. The objective is to maintain meaningful genetic material while ensuring a valid and complete tour in the offspring as shown in Fig. 4.

Hybrid Local Search and Genetic Algorithm, HLGA

Input:

- *n_points*: total number of cities.
- *x, y*: coordinates of the cities.
- *N*: the total number of initial solutions, set to 10% of *n_points* are generated via the local search procedure, and the remaining solutions are generated by the genetic algorithm (GA).
- *path*: a vector representing the visiting sequence of cities.
- *distances*: pairwise Euclidean distances between cities.

Output:

- Fitness values of solutions obtained by 2-opt, 3-opt, and hybrid Genetic Algorithm.
- *The best solution*.

1: For each of the *N* initial solutions:
  1.1 Generate a solution at randomly.
  1.2 Each solution applies 2-opt and 3-opt local searches to optimize its subtours.
2: Hybridize with the genetic algorithm (GA):
  2.1 Generated *custom population* via random permutations.
  2.2 Merge the high-quality local optima from step 1.2 with the GA-generated population to form an improved *good initial solution*.
  2.3 Evaluate the fitness of each *good initial solution* in this enhanced initial set.
3: Apply Genetic Operations for *good initial solution*:
  3.1 Selection: Choose parents using tournament or roulette selection.
  3.2 Crossover: Apply Order-Based Crossover (OX) to generate offspring.
  3.3 Mutation: Perform inversion or swap mutation with adaptive probability.
4: Obtain the result: Iterate GA until termination criteria met and return the *best solution* found during the evolutionary process.

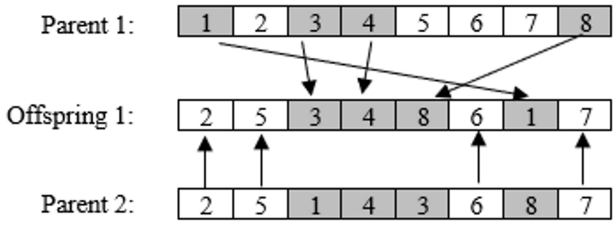

***Fig. 4**: Order-Based Crossover (OX).*

***Table 2**: Quality of HLGA compared to their BKS for the 27 benchmark instances.*

Number	Description
27	datasets better than 3-opt
6	datasets equal to BKS
5	datasets better than BKS
15	datasets with RE between 0% and 3%

4. RESULTS AND DISCUSSIONS

This study investigates performance enhancement strategies for solving the TSP, using R version 4.5.0 and RStudio version 2024.12.1+563 on Windows 10. A total of 27 benchmark instances were selected from the TSPLIB repository [22], with problem sizes 30 to 200 cities.

In all experiments, the HLGA employed a population size $P = 200$ and a maximum generation limit $G_{max} = 5,000$. A hybrid initialization was adopted in which 10% of the population ($N = \lceil 0.1 \times \text{n\_points} \rceil$) consisted of elite seeds generated by 2-opt and 3-opt local searches applied to random tours, while the remaining 90% were created through uniform random permutations to preserve diversity. The GA used an order-based crossover (OX) operator with crossover probability (*pcrossover*) = 0.8, an inversion mutation operator with mutation probability (*pmutaion*) = 0.2, a fitness-proportionate (roulette-wheel) selection mechanism, and an elitism policy that preserved the best individual each generation. A convergence threshold of 300 consecutive stagnant generations was applied as the termination condition. In addition, for each experiment, all algorithms were executed 10 times independently for each problem, and results of the tests were presented based on the best, mean, standard deviations and relative error. The experimental results, together with the results of the other algorithm, are shown in Table 2, Quality of HLGA is compared to their BKS for the 27 benchmark instances. Tables 3 – 5 shows details of tests with our model, HLGA.

In this study, firstly 27 problems were selected from TSPLIB [22] for comparison with the 3-opt local search and HLGA model. The experimental summary results are shown in Table 2. For the HLGA, it summarizes the superior performance of the HLGA model compared to the 3-opt method across all 27 datasets. HLGA identified the Best-Known Solution (BKS) for 6 datasets (ch130, kroD100, kroE100, Pr76, pr124, and rd100) with a Relative Error (RE) of 0.0%. In 5 datasets (kroA150,

***Table 3**: Performance comparison between HLGA and 3-opt local search.*

Instance	N	BKS	3-opt				**HLGA**			
			Best	Avg	SD	RE(%)	Best	Avg	SD	RE(%)
berlin52	52	7542	7891	8350.31	293.02	4.62	**7544**	7586.71	127.04	**0.02**
bier127	127	118282	121016	124720.30	1887.39	2.31	**118578**	120479.00	1003.05	0.25
ch130	130	6110	6325	6533.51	156.27	3.51	**6110**	6237.77	79.79	**0.00**
ch150	150	6528	6707	6963.96	117.90	2.74	**6552**	6665.22	94.50	0.36
eil51	51	426	440	459.83	12.74	3.28	**428**	435.92	6.93	0.46
eil76	79	538	559	586.19	17.53	3.90	**545**	553.98	4.96	1.30
eil101	101	629	656	684.47	16.00	4.29	**640**	654.55	8.24	1.74
kroA100	100	21282	21738	23639.10	1274.77	2.14	**21285**	21655.44	325.04	**0.01**
kroA150	150	26524	27704	28162.46	617.19	4.44	**26127**	27051.89	279.10	**-1.49**
kroA200	200	29368	30445	31694.05	1087.97	3.66	**29481**	30037.51	411.36	0.38
kroB100	100	22140	22792	23987	855.27	2.94	**22139**	22423.54	243.63	**0.00**
kroB150	150	26130	27861	28265	206.12	6.62	**26127**	26411.65	300.43	**-0.01**
kroB200	200	29437	30164	31453	1092.5	2.46	**29700**	30186.90	321.99	0.89
kroC100	100	20749	21169	22759.03	991.54	2.02	**20750**	20937.83	178.07	**0.00**
kroD100	100	21294	21663	22984.96	1015.22	1.73	**21294**	21543.78	177.00	**0.00**
kroE100	100	22068	22379	24335.61	1418.37	1.40	**22068**	22300.22	139.33	**0.00**
rat99	99	1211	1270	1302.94	33.14	4.87	**1220**	1252.54	17.69	0.74
rat195	195	2323	2409	2465.29	47.87	3.70	**2384**	2416.73	19.91	2.62
pr76	76	108159	110221	114424.24	3531.93	1.90	**108159**	108855.44	561.99	**0.00**
pr107	107	44303	44619	46466.50	1563.08	0.71	**44301**	44444.22	107.52	**0.00**
pr124	124	59030	60772	65467.64	2858.81	2.95	**59030**	59927.52	503.68	**0.00**
pr136	136	96772	98053	103809	322657	1.32	**97099**	99627.51	1168.95	0.33
pr144	144	58537	60795	64005.83	2746.50	3.85	**58535**	58826.32	382.45	**0.00**
lin105	105	14379	14808	15739.01	650.33	2.98	**14383**	14612.56	218.02	**0.02**
rd100	100	7910	8287	8651.87	234.39	4.76	**7910**	8019.58	102.41	**0.00**
st70	70	675	692	727.70	24.87	2.51	**677**	688.03	8.08	0.29
Oliver30	30	420	423	457.68	22.97	0.71	**423**	423.00	0.00	0.71
Sum Relative Error						**83.91**				**9.22**

kroB100, kroB150, pr107, and pr144), HLGA obtained objective values below the reported BKS, demonstrating the benefit of locally refined initial solutions. Using 2-opt for speed and 3-opt for stronger escape capability creates a balanced, high-quality initial population for the Genetic Algorithm. This hybrid refinement produces diverse elite seeds across runs, improving exploration and reducing premature convergence. For another 15 datasets, the *RE* was in the range of 0% < *RE* ≤3%. Detailed results are presented in Table 3.

Relative Error (*RE*) was determined by comparing the solution quality obtained by the algorithm with the known optimal solution or the best-known solution for the specific TSP instance. The formula for *RE* is as follows:

$$RE\ (\%) = \frac{S\ -\ BKS}{BKS} \times 100 \qquad (2)$$

Where: *s* represents the best solution obtained from 10 runs of the algorithm, and *BKS* denotes the optimal or best-known solution.

The first column of Table 3 gives the TSPLIB dataset name and the second and third columns are number of cities and length for the best-known solution (BKS). The next 4 - 7 columns contain the results of 3-opt that contain best value (Best), mean value (Avg), standard deviation value (SD), and relative error value (RE). The last four columns show the results of this study, HLGA is the result of 10 runs.

Furthermore, HLGA exhibited lower average solution values and standard deviations than the 3-opt method. This performance advantage is reinforced by the aggregate RE across all 27 instances: 9.22 for HLGA compared to 83.91 for 3-opt. Table 3 provides detailed results for each instance. As shown in Fig. 5, RE comparisons of HLGA and 3-opt on TSP datasets, RE of 0 for an instance indicates that the obtained solution corresponds to the Best-Known Solution (BKS).

The comparative performance of the proposed HLGA is shown in Table 4 for 27 instances. It is observed that HLGA provides better solutions than the GGSC-SSA proposed by Wu et al. [3], SCGA proposed by Deng et al. [4], IMode/J2000 proposed by Krömer & Uher [5], ACO-DSA proposed by Hao et al. [6], and RL-SA proposed by Gil-Gala et al. [8] on all the test instances considered. The HLGA produced the most prominent solutions, that achieved the Best-Known Solution (BKS) for 6 instances (ch130, kroD100, kroE100, pr76, pr124, and rd100); yielded solutions with objective values numerically lower than the BKS (indicating better performance) for 5 instances (kroA150, kroB100, kroB150, pr107 and pr144). GGSC-SSA model, achieved the Best-Known Solution (BKS) for 3 instances (berlin52, eil51, and rat99) and results better than BKS for 1 instance (kroA100). ACO-DSA model achieved the lower than BSK for 1 instance

***Table 4**: Performance comparison between HLGA and other five algorithms.*

Instance	BKS	GGSC-SSA[3]	SCGA[4]	IMODE/J2020[5]	ACO-DSA[6]	RL-SA[8]	**HLGA**
berlin52	7542	**7542***	7544.3	7897.6	7544.4	7598	7544
bier127	118282	-	118293.5	148729.8	118703.6	123771	118578
ch130	6110	6115.25	6183	8611.5	6117.5	6722	**6110***
ch150	6528	6533	-	9943.5	6557.3	6788	6552
eil51	426	**426***	428.8	444.2	428.9	434	428
eil76	538	539.79	547.1	588.3	548.3	565	545
eil101	629	638	-	739	656.0	672	640
kroA100	21282	**20989.04****	21285.4	27313.3	21381.8	21683	21285
kroA150	26524	-	-	41961.5	-	29728	**26127****
kroA200	29368	29507.35	29533.0	55752.4	30428.7	32713	29481.82
kroB100	22140	22152	-	27925	-	23412	**22139****
kroB150	26130	-	-	40811.1	-	27564	**26127****
kroB200	29437	29678.9	-	56017		34281	29700
kroC100	20749	21346.53	-	27081.5	-	21545	20750
kroD100	21294	22138.53	-	27382.6	-	22615	**21294***
kroE100	22068	-	-	27550.2	-	22712	**22068***
rat99	1211	**1211***	-	1481	-	1271	1220
rat195	2323	-	-	-	-	2568	2384
pr76	108159	109170.3	-	118737.9	108159.4	111085	**108159***
pr107	44303	-	44301.6	58637	**44301.7****	45162	**44301****
pr124	59030	59607.74		88953.5	-	61541	**59030***
pr136	96772	-	97102.3	138015	97367.8	107784	97099
pr144	58537	58862.09	-	102914.2	-	60763	**58535****
lin105	14379	14543.5	-	18497.6	-	15579	14383
rd100	7910	7951.28	-	10003.9	-	8439	**7910***
st70	675	676.96	-	755.5	677.1	710	677
Oliver30	420	-	423.7	-	-	-	423

*BKS; **< BKS

***Table 5**: Performance comparison between HLGA and other three algorithms.*

Instance	BKS	GGSC-SSA [3]	IMODE/J2020 [5]	RL-SA [8]	**HLGA**
berlin52	7542	**7542***	7897.6	7598	7544
ch130	6110	6115.25	8611.5	6722	**6110***
ch150	6528	**6533**	9943.5	6788	6552
eil51	426	**426***	444.2	434	428
eil76	538	**539.79**	588.3	565	545
eil101	629	**638**	739	672	640
kroA100	21282	**20989.04****	27313.3	21683	21285
kroA200	29368	29507.35	55752.4	32713	**29481.82**
kroB100	22140	22152	27925	23412	**22139****
kroB200	29437	**29678.9**	56017	34281	29700
kroC100	20749	21346.53	27081.5	21545	**20750**
kroD100	21294	22138.53	27382.6	22615	**21294***
rat99	1211	**1211***	1481	1271	1220
pr76	108159	109170.3	118737.9	111085	**108159***
pr124	59030	59607.74	88953.5	61541	**59030***
pr144	58537	58862.09	102914.2	60763	**58535****
lin105	14379	14543.5	18497.6	15579	**14383**
rd100	7910	7951.28	10003.9	8439	**7910***
st70	675	**676.96**	755.5	710	677

*BKS; **< BKS

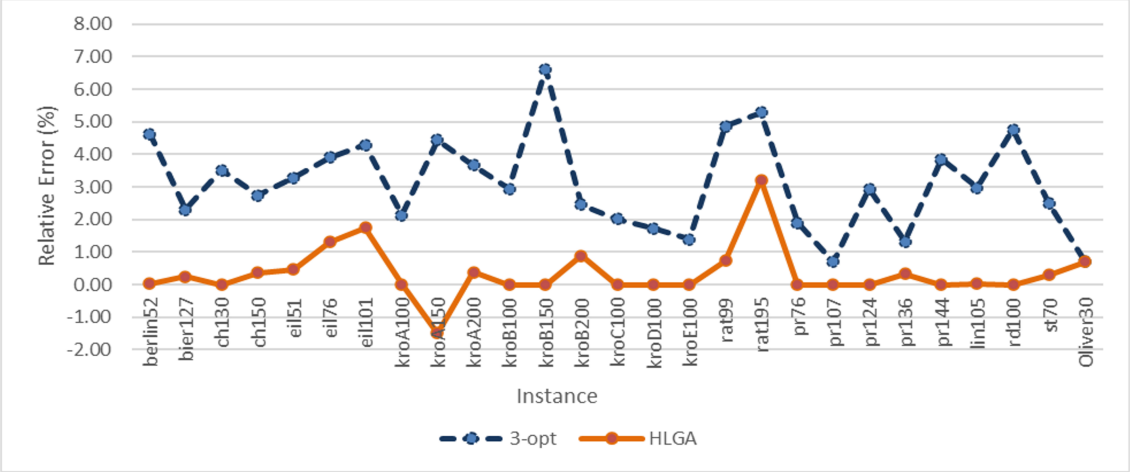

***Fig. 5**: RE comparisons of HLGA and 3-opt on TSP datasets.*

(pr107), and SCGA model and IMODE/J2020 were not found.

Table 5 shows the comparative performance analysis that was conducted against three other algorithms—GGSC-SSA [3], IMODE/J2020 [5], and RL-SA [8]—focusing on 19 instances to ensure a clear evaluation. The HLGA model achieved the Best-Known Solution (BKS) for 5 instances (ch130, kroD100, pr76, pr124, and rd100); yielded solutions with objective values numerically lower than the BKS (indicating better performance) for 2 instances (kroB100 and pr144); and provided the closest solutions to the BKS for 3 instances (kroA200, kroC100, and lin105). GGSC-SSA [3] achieved the BKS for 3 instances (berlin52, eil51, and rat99); obtained a solution with an objective value lower than the BKS for 1 instance (kroA100); and delivered the closest solutions to the BKS for 5 instances (ch150, eil76, eil101, kroB200, and st70).

A comparable detailed breakdown of results was not found for IMODE/J2020 [5] and RL-SA [8] on this specific set of 19 instances.

5. CONCLUSION AND FUTURE WORK

In this paper, a hybrid heuristic approach for the traveling salesman problem based on the genetic algorithm, HLGA was investigated. Considering the characteristics of TSP instances, this paper focused on generating high-quality and diverse initial solutions to enhance the search for the global optimum. The HLGA was benchmarked against five state-of-the-art algorithms on 27 instances from the TSPLIB [26]. Experimental results demonstrate that the proposed strategy of employing local search to refine the initial population significantly accelerates the evolutionary optimization process. Consequently, the HLGA proves to be a highly competitive approach for solving TSP problems. While it is acknowledged that local search strategies are commonly integrated within mutation operations to prevent premature convergence to local optima, this study highlights an alternative benefit by emphasizing the importance of the initial solution. The approach focuses on obtaining good solutions close to the target value first, and then through targeted local refinement and diversification, the subsequent search process is more effectively guided towards achieving the global optimal solution using GA methods.

The subsequent search process involves iterative improvements until better solutions are found, which requires more time. Therefore, this study utilizes a limited number of solutions ($N = 10\%$) to more effectively guide the search towards achieving the global optimal solution using GA methods.

Although this study primarily focused on evaluating the solution quality of the proposed HLGA across multiple TSPLIB benchmark instances, the examination of computational runtime was not addressed in detail in the present work. Since runtime efficiency constitutes an important dimension of algorithmic performance, future research will incorporate a comprehensive analysis of execution time, together with a comparative assessment between the HLGA and the conventional 3-opt heuristic. These experiments will be conducted under controlled random seeds, enabling paired measurements of runtime and solution quality and thereby facilitating a more rigorous evaluation of the time–quality trade-off.

ACKNOWLEDGEMENT

This work was supported by Rajamangala University of Technology Lanna, Chiang Mai, Thailand. The author gratefully acknowledges the scholarship support provided by the Office of the National Science and Technology Development Agency (formerly known as NSTDA), Thailand.

REFERENCES

[1] G. K. D. Saharidis, "Review of Solution Approaches for the Symmetric Traveling Salesman Problem," *International Journal of Information Systems and Supply Chain Management*, vol. 7, no. 1, pp. 73-87, Jan. 2014.

[2] P. C. Pop, O. Cosma, C. Sabo, and C. P. Sitar, "A comprehensive survey on the generalized traveling salesman problem," *European Journal of Operational Research*, vol. 314, no. 3, pp. 819-835, May. 2024.

[3] C. Wu, X. Fu, J. Pei, and Z. Dong, "A novel sparrow search algorithm for the traveling salesman problem," *IEEE Access*, vol. 9, pp.153456-153471, Nov. 2021.

[4] Y. Deng, J. Xiong, and Q. Wang, "A hybrid cellular genetic algorithm for the traveling salesman problem," *Mathematical Problems in Engineering*, vol. 2021, pp. 1-16, Feb. 2021.

[5] P. Krömer and V. Uher, "Optimization of real-world supply routes by nature-inspired metaheuristics," in *Proceeding of the IEEE Congress on Evolutionary Computation (CEC)*, Padua, Italy, Sep. 2022, pp. 1-10.

[6] T. Hao, W. Yingnian, Z. Jiaxing, and Z. Jing, "Study on a hybrid algorithm combining enhanced ant colony optimization and double improved simulated annealing via clustering in the traveling salesman problem," *PeerJ Computer Science*, vol. 9, e1609, Oct. 2023.

[7] F. J. Gil-Gala, M. Durašević, M. R. Sierra, and R. Varela, "Evolving ensembles of heuristics for the travelling salesman problem," *Natural Computing*, vol. 22, pp. 671-684, Oct. 2023.

[8] L. Hong, Y. Liu, M. Xu, and W. Deng, "Combining deep reinforcement learning with heuristics to solve the traveling salesman problem," *Chinese Physics B*, vol. 34, no. 1, p. 018705-1-11, Jan. 2025.

[9] D. C. Diana, and R. Hema, "Swarm Intelligence Based MMSE Frequency Domain Equalization for MIMO Systems," *ECTI Transactions on Electrical Engineering, Electronics, and Communications,* vol. 21, no. 2, pp. 1-10, Jun. 2023.

[10] S. Kumari and R. K. Mandal, "Whale Optimization Algorithm Based Closed-Loop Control Z-source Inverter for Wind Energy Application," *ECTI Transactions on Electrical Engineering, Electronics, and Communications*, vol. 22, no. 2, pp. 1-12, Jun. 2024.

[11] W. Teekeng and A. Thammano, "Modified genetic algorithm for flexible job-shop scheduling problems," *Procedia Computer Science*, vol. 12, 2012, pp. 122-128.

[12] N. Kafakthong and K. Sinapiromsaran, "Near-Optimal Traveling Salesman Solution with Deep Attention," *International Journal of Advanced Computer Science & Applications*, vol. 15, no. 12, pp. 954-963, 2024.

[13] J. Sui, S. Ding, R. Liu, L. Xu, and D. Bu, "Learning 3-opt heuristics for traveling salesman problem via deep reinforcement learning," in Proceedings of The 13th Asian conference on Machine Learning Research vol. 157, 2021, pp. 1301-1316.

[14] E. O. Asani, A. E. Okeyinka, S. A. Ajagbe, A. A. Adebiyi, R. O. Ogundokun, T. S. Adekunle, P. Mudali, and M. O. Adigun, "A Novel Insertion Solution for the Travelling Salesman Problem," *Computers, materials and continua*, vol. 79, no. 1, pp. 1581-1597, 2024.

[15] V. Pacheco-Valencia, J. A. Hernández, J. M. Sigarreta, and N. Vakhania, "Simple Constructive, Insertion, and Improvement Heuristics Based on the Girding Polygon for the Euclidean Traveling Salesman Problem," *Algorithms*, vol. 13, no. 1, Dec. 2020.

[16] D. E. Knuth, *The art of computer programming*, Volume 3: Sorting and searching, Addison-Wesley, Reading, MA, 1973.

[17] G. A. Croes, "A method for solving traveling-salesman problems," *Operations research*, vol. 6, no. 6, pp. 791-812, Nov. 1958.

[18] S. Kefi, N. Rokbani, P. Krömer and A. M. Alimi, "Ant supervised by PSO and 2-Opt algorithm, AS-PSO-2Opt, applied to Traveling Salesman Problem," in *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Budapest, Hungary, 2016, pp. 004866-004871.

[19] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, Reading, MA, 1989.

[20] A. Thammano, and W. Teekeng, "A modified genetic algorithm with fuzzy roulette wheel selection for jobshop scheduling problems." *International Journal of General Systems*, vol. 44, no. 4, pp. 499-518, 2015.

[21] Z. Lin and J. Li, "A hybrid genetic algorithm with cycle reassembly for solving colored traveling salesman problems," *Journal of King Saud University – Computer and Information Sciences*, 2023.

[22] A. J. Ibada, B. Tűű-Szabó, and L. T. Kóczy, "The Circle Group Heuristic to Improve the Efficiency of the Discrete Bacterial Memetic Evolutionary Algorithm Applied for TSP, TRP, and TSPTW," *Symmetry*, vol. 17, no. 10, p. 1683, 2025.

[23] D. R. Singh, "Using Group Theory to Generate Initial Population for a Genetic Algorithm for Solving Traveling Salesman," in *Genetic Algorithms Theory, Design and Programming*, IntechOpen, 2023.

[24] Y. Wang, Y. Chen, and Y. Lin, "Memetic algorithm based on sequential variable neighborhood descent for the minmax multiple traveling salesman problem," *Computers & Industrial Engineering*, vol. 106, pp. 105–122, 2017.

[25] A. M. H. Al-Ibrahim, "Solving Traveling Salesman Problem (TSP) by Hybrid Genetic Algorithm (HGA)," *International Journal of Advanced Computer Science and Applications*, vol. 11, no. 6, 2020.

[26] G. Reinelt, "TSBLIB - A traveling salesman problem library", *ORSA journal on computing*, vol. 3, no. 4, pp. 267-384, Nov. 1991.

Wannaporn Teekeng works as a lecturer in Business Information System, Rajamangala University of Technology Lanna, Chiang Mai, Thailand.