IAM-TSP: Iterative Approximate Methods for Solving the Travelling Salesman Problem

Esra'a Alkafaween[1], Samir Elmougy[2], Ehab Essa[3], Sami Mnasri[4], Ahmad S.Tarawneh[5], Ahmad Hassanat[6]
E-Learning and Education Resources Center, Mutah University, Karak, Jordan[1]
Department of Computer Science, Mansoura University, Mansoura, Egypt[2, 3]
Department of Computer Science, University of Tabuk, Tabuk, Saudi Arabia[4]
Computer Science Department, Mutah University, Karak, Jordan[5, 6]

***Abstract*—TSP is a well-known combinatorial optimization problem with several practical applications. It is an NP-hard problem, which means that the optimal solution for huge numbers of examples is computationally impractical. As a result, researchers have focused their efforts on devising efficient algorithms for obtaining approximate solutions to the TSP. This paper proposes Iterative Approximate Methods for Solving TSP (IAM-TSP), as a new method that provides an approximate solution to TSP in polynomial time. This proposed method begins by adding four extreme cities to the route, a loop, and then adds each city to the route using a greedy technique that evaluates the cost of adding each city to different positions along the route. This method determines the best position to add the city and the also the best city to be added. The resultant route is further improved by employing local constant permutations. When compared to existing state-of-the-art methods, our experimental results show that the proposed method is more capable of producing high-quality solutions. The proposed approach, with an average approximation of 1.09, can be recommended for practical usage in its current form or as a pre-processing step for another optimizer.**

Keywords*—Greedy algorithms; TSP; NP-Hard problems; polynomial time algorithms; combinatorial problems, optimization methods

I. INTRODUCTION

At the beginning of the seventeenth century, Thomas Penynington and William Hamilton modeled the first mathematical problem corresponding to the Traveling Salesman Problem (TSP). This problem represents a game in which the winner connects twenty points by moving from one point to another, following some precise paths. This game/problem is called Hamilton Circuit Theory [1]. Afterward, in graph theory, TSP becomes a problem of identifying the optimal (shortest) Hamiltonian cycle visiting a set of cities (points) using a matrix of distances between the cities. TSP is one of the classic problems of combinatorial optimization, and it is widely investigated and considered a standard for assessing the performance of computational methodologies and algorithms.

The principal objective of TSP is that the salesman visits all the cities in the minimal tour and then returns to the starting point with the assumption that the distances between the cities are known and the necessity of visiting each city is only once. Despite high-size instances of TSP being resolved in the literature, TSP is proven to be NP-hard even for small-size instances [2].

The theoretical and practical relevance of the TSP problem comes from the fact that numerous applicable real-world and engineering problems can be solved by adapting the TSP solutions, such as circuit design [3], scheduling [4], and DNA [5, 6], in addition to logistics and transportation and even space exploration missions [7]. Therefore, the study of TSP has significant theoretical and practical value, leading to cost savings and other benefits.

Moreover, the application fields of TSP include any problem involving the search for the shortest paths. These fields vary in domains from big data classification [8, 9, 10], deployment and routing of IoT networks [11, 12, 13, 14, 15, 16], financial prediction [17, 18], image processing [19], and [20], computer vision [21], [22], and [23]. TSP can be defined by finding a Hamiltonian cycle that visits each vertex precisely once, with the least amount of total weight feasible, given a full undirected weighted graph G = (V, E) with vertex set V and edge set E. The total weight of a cycle is equal to the weights of each of its individual edges. The book in [24] presents a comprehensive mathematical and theoretical analysis of the calculability and complexity of the TSP. One of the critical issues of methods of resolution of TSP is the enhancement of the accuracy of the algorithm to rapidly find the optimal or near-optimal solutions.

Indeed, heuristics and meta-heuristics are the most successful methods used to resolve the TSP [25, 26, 27]. In this regard, Genetic Algorithms (GA), Ant Colony Optimization (ACO), Simulated Annealing (SA), and Particle Swarm Optimization (PSO) are among the most commonly used algorithms to resolve TSP. GA was successfully used for problems involving global search due to its rapid convergence and fast search process. However, it has some issues and has poor performance when achieving the local search. ACO is another robust optimizer with a high-resolution capacity. However, its convergence performance highly depends on the initial parameters, mainly the appropriate initial quality of the pheromone. PSO is characterized by its ease of use since it has a few numbers of initial parameters to set. However, other optimizers give better results for many TSP test problems. SA is another optimizer known for its capacity to avoid local optima and is often hybridized with other algorithms for this capacity.

However, like with ACO, the initial parameter values have a significant impact on the search process. Despite substantial research on TSP, its combinatorial nature suggests that there is still opportunity for development in this field, motivating the work of this paper. TSP is inextricably tied to handling other combinatorial problems with comparable characteristics, such as the Traveling Salesman Problem with Time Windows (TSPTW), Vehicle Routing Problem (VRP), and Capacitated Vehicle Routing Problem (CVRP). TSP approaches and algorithms are frequently used as a foundation for addressing these related optimization challenges. As a result, progress in TSP solving may provide useful insights and tactics relevant to a broader class of combinatorial problems in logistics, transportation, and network optimization. Because these challenges are interdisciplinary, knowledge and approaches can be transferred across fields, creating a more thorough grasp of combinatorial optimization.

The major contribution of this paper lies in presenting an Iterative Approximate Method for Solving TSP (IAM-TSP), which provides a polynomial-time approximate solution to the TSP. The proposed method begins by including four extreme cities along the path. Following that, a loop successively adds each city to the route by calculating the cost at various points along the route. The approach considers each city to add and finds the most suitable location to put it, the one that minimizes the total cost. This method is improved further by using local constant permutations on the output of IAM-TSP, which considerably improves the final route, referred to as IAM-TSP+.

We evaluated the proposed algorithms against standard TSP datasets to determine how they performed in terms of key performance measures. The proposed IAM-TSP performed nearly identically to some of the typical TSP approximation algorithms given in the results section, but the proposed IAM-TSP+ surpassed all approximation methods compared on the majority of TSP instances.

The rest of the paper is organized as follows. Section II illustrates and discusses the advantages and drawbacks of the recent most relevant studies resolving the TSP. Section III identifies and investigates the proposed methodology for resolving the TSP. Section IV presents the experimental setup and the results, Section V concludes the study.

II. Related Work

Given TSP's extensive history, its cutting-edge landscape is extremely diverse. To solve the TSP, several techniques, paradigms, and approximations algorithms have been used. Heuristics, for example, are a type of approximation method aimed to finding near-optimal solutions to NP-hard problems in polynomials time. One popular and simple method for building a TSP tour involves starting the tour at any node, traversing minimum-cost arcs to each successive node until all nodes are visited, and then returning to the starting node to finish the tour. This method is known as the Nearest Neighbor heuristic [28].

There are also many popular methods, such as: the 2-Opt heuristic [29], Farthest Insertion algorithm [30], Nearest Insertion algorithm [30], Cheapest Insertion algorithm [30], Arbitrary Insertion algorithm [31], Repetitive Nearest Neighbor algorithm [32], Concave hull with heuristics, and Concave Hull No Heuristic [33]. In what follows, the main recent studies proposing TSP resolution methodologies are investigated: The study in [33] introduces a concave hull-based algorithm to resolve the Euclidean symmetric TSP by establishing concentric hulls and then merging them in one tour. Two novel metrics are suggested: the “Average Waiting Distance” and the “min AWD” of a tour. The results show that an optimal AWD does not guarantee the optimal tour.

In study [34], the authors enhanced the accuracy of TSP solutions for numerous sizes of the problem. The used algorithm is a modified ACO with a better convergence during the TSP search process. To prevent trapping into local optima, this algorithm decreases the high concentration of pheromone during the route selection. The diversity in the algorithm is ensured using entropy weighted learning. According to the results, this work improved ACO and solved TSP better than the standard ACO.

A new variant of TSP, called TSPJ, which includes the schedule of jobs in a set of positions, was introduced in study [35]. Since in TSP, the transport time is longer than the operation time, the considered objective in TSPJ is to minimize the make-span, equal to the needed time to achieve the longest job. The resolution of the TSPJ involves four local search methods. Using the CPLEX system, the results indicated that the solutions given by the four used heuristics are too close to the optimal (a gap less than 6%). In the same regard, the study in [36] aims to resolve the TSPJ. Applicative and practical contexts of TSPJ had been discussed, as well as the parameters specific to TSPJ such as the completion time, configuration time, and resource variation.

In study [37], the authors used the aim was to resolve a TSP variant called the multiple TSP (mTSP) problems using a hybrid algorithm. The latter relies on an EAX heuristic to optimize the intra-tour and a tabu search neighborhood search to optimize the inter-tour. The objectives of the problem were the minimization of both the longest path and the total traveled distance. To reduce the neighborhood search time, a reduction approach is proposed to avoid computing the nonpromising possible solutions. The experiments involve a comparison with five approaches tested on 41 known TSP test problems and 36 new large-size ones. However, the other variants of TSP were not assessed, and comparisons with other classes of heuristics were not achieved.

The study in [38] suggests a new seriation strategy named “tree-penalized Path Length” (tpPL). Data seriation is a famous problem in data analysis. Itis the process of sequencing and ordering data according to their similarity. TSP in this study is considered a seriation method. The goal is to linearly order the data using simultaneously the TSP, tpPL, and optimal leaf order (OLO) methods. Optimal paths are transferred from TSP to OLO. In terms of computational complexity, TSP and tpPL have the same order of complexity. Hence, TSP heuristics may be used to resolve the tpPL. Tested on more than forty datasets, the tpPL has a better performance than TSP and OLO with the same computational complexity. The study in [39] introduces a compression-based TSP heuristic for data micro-aggregation.

Micro-aggregation is a method for disrupting and aggregating personal data using the concept of k-anonymity. By simultaneously considering the respect for privacy and the usefulness of data, the introduced TSP heuristic was the most efficient in solving the problem of micro-aggregation. In contrast, heuristics relying on TSP encrust scalability issues. Unlike other heuristics, the algorithm proposed in this study can reduce the execution time of the TSP. The tests carried out on small and medium-sized data affirm the trade-off between computation time and the rate of loss of micro-aggregation information.

In study [40], a new problem, named the traveling thief problem (TTP), is proposed by combining the knapsack problem (KP) and the TSP. The PTT resolution method relies on sequences selection heuristics. A set of selection operators/thieves are involved to select the cities/objects to progressively create the tour.

The study in [41] introduces two new TSP versions. Named pollution TSP(PTSP) and energy minimization TSP (EMTSP), the new versions add environmental constraints to the TSP. The aim of PTSP is to reduce fuel consumption, and carbon emissions. The aim of EMTSP is to reduce the cost of a trip according to the distance and the carried load. The method of resolution of the two TSP variants, MILP-GA-LS, relies on a mixed integer linear programming model to identify initial possible solutions, then multi-operator GA to enhance the found solutions. Afterward, an iterative local search algorithm was used to enhance the solutions. However, numerous constraints were not considered such as the time-window of customers. Moreover, an issue arises concerning the complexity of the introduced approach, since it was proven that, for small-size instances of PTSP and EMTSP, exact methods give better results than the MILP-GA-LS.

In study [42], a new ACO variant is proposed to solve the TSP. Named DAACO; this algorithm dynamically changes the number of ants to avoid falling into local optima and to prevent long time of convergence. Besides, DAACO uses a local selection process to enhance the quality of ants and the needed time for the search process. Among the used twenty TSPLIB test problems, all the DAACO solutions were optimal except one. These results confirm the advantageous quality of solutions and time of convergence of DAACO compared to other optimizers.

In study [43], a hybrid Ant Colony (AC)-Tabu Search (TS)-Firefly Algorithm (FA) called ACTS-FATS is proposed. The hybridization avoids the probability of premature convergence, then, reduces the chance of trapped in local optima. Tested with the TSPLIB95, the hybridization does not generate additional execution time compared to AC, TS and FA.

In study [44], Dhouib-Matrix, a column-row method, is proposed to resolve polynomial time TSP. The process of this method is as follows: after defining the distance matrix, a start position is selected to choose rows. Then, columns are discarded, and the route is transformed to a tour. The advantage of the introduced Dhouib Matrix is that it needs only n iterations to find the route between n cities.

In study [45], a comparative study is proposed to discuss the recent algorithms and methodologies used to resolve the TSP using metaheuristics. The focus is set on the numerous versions of the BA, FA and PSO optimizers.

In study [46], a new version of the TSP, called TSP-D is proposed. The latter is a classic TSP that involves a truck and a drone (Unmanned Aerial Vehicles (UAV)). In TSP-D, it is assumed that UAV has low battery capacities and can transport cargo per flight. The issue is to determine which UAV and which truck should serve which client. The tests demonstrate that the time of service can be reduced by using the UAVs with distinct speeds according to the cargo weight. However, the used samples (number of clients) are between 30 and 60, which makes the method valid only for small-size TDP-D instances.

The authors in [47] propose an algorithm to minimize travel costs and maximize the overall profit. Simulated annealing (SA) and genetic algorithm (GA) with dedicated mutation operators were used. A concept of tour plots is used to generate the final solutions. In terms of computation time, the SA is better than the GA.

In study [48], Qi-ACO, an ACO based on quantum computing, is developed to resolve the four-dimensional TSP (4DTSP). Fuzzy type-2 variables are used due to the uncertain aspect of the investigated problem of travel emissions and costs. The 4DTSP is characterized by the existence of numerous paths and conveyances between the cities. A process is implemented in Qi-ACO for generating qubits considering the constraints of carbon emission, cost, and time. The initialization and update of pheromone is based on qubit. A faster computation is achieved in Qi-ACO due to the quantum calculations. Performed statistical tests to confirm the performance of the introduced approach. However, numerous constraints are not considered such as vehicle speed and route selection. The readers are referred to [49] for more comprehensive methodologies on modeling and designing greedy heuristic methods to resolve the TSP.

III. The Proposed Iterative Approximate Methods for Solving TSP (IAM-TSP)

The IAM-TSP is proposed in this paper as an approximation to the TSP, which has a greedy algorithmic nature. It begins by selecting four cities from the input set to represent the east, north, west, and south most positions. These cities are added to the route list in the following order: east, north, west and south, followed by a return to the beginning point (east). The algorithm then enters a loop, adding each remaining city to the path one by one. The method calculates the cost of adding a city to various points along the route throughout each iteration. It carefully investigates all feasible positions for each city, picking the best position and city at the lowest cost. Finally, this approach computes the route's total cost and outputs the final result.

The cost is calculated using the Euclidean distance between the consecutive cities on the route. Algorithm 1 shows the pseudocode processes of the proposed IAM-TSP. Fig. 1 illustrates the progress of the IAM-TSP solution of the Rat195 LIBTSP real-world problem [50]. Because the TSP is a

minimization problem, the term "best" refers to the smallest value throughout this paper, i.e., minimum route.

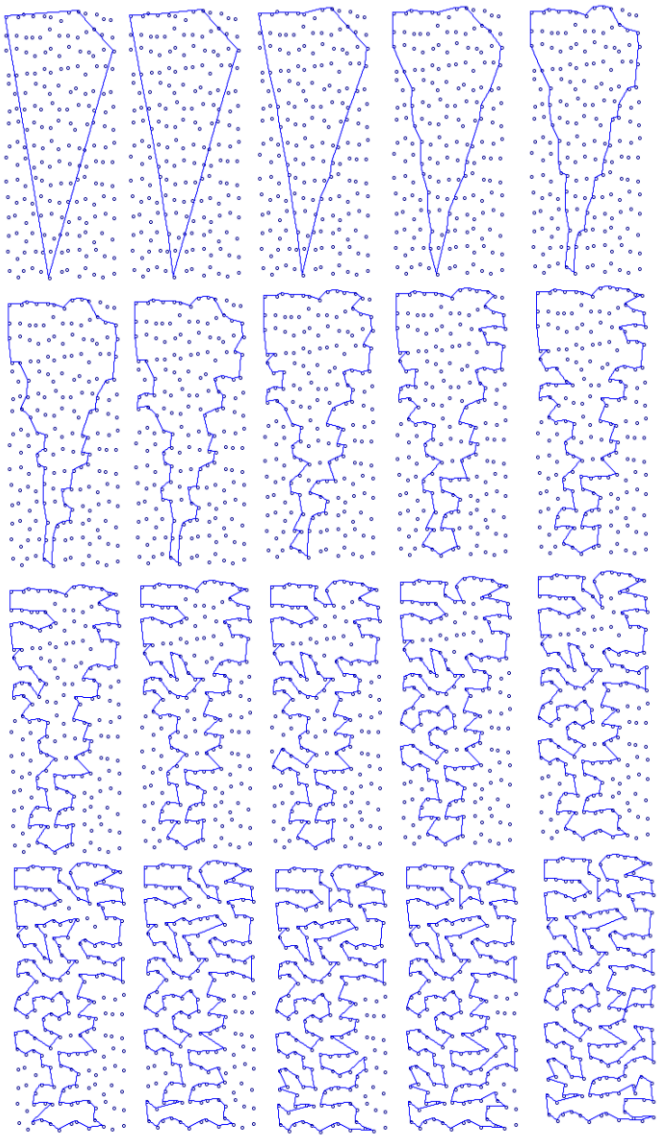

Fig. 1. The progress of the IAM-TSP solution of the Rat195 LIBTSP real-world problem, the progress is made from left to right and from top to bottom, and the results are displayed after ten iterations from the previous state.

According to Algorithm 1, the proposed IAM-TSP has a time complexity of $O(n^2)$, where n is the number of cities in the Cities x y list. This is because, in the while loop, the algorithm finds the best position to put each city from the Open list into rout and checks every feasible solution, which takes O(n) time. This operation is repeated until Open is empty, which means that in the worst scenario, the while loop runs n times, giving a total time complexity of $O(n^2)$.

The space complexity of the solution is $O(2n + n^2)$, where n is the number of cities in the Cities xy list. This is because we use two lists rout and open of size n, which take O(2n) space, and the quadratic space comes from the space reserved by the Possible Solutions matrix, whose size is n rows, each hosting n cities. The rest of the data structures used in the solution take O(1) or O(n) space, so the overall space complexity can be asymptotically approximated to $O(n^2)$. However, if we establish a square matrix storing the distances between each city and the others, space complexity stays asymptotically quadratic too; this is not done in this study, but it is a typical approach to removing the burden of distance computation. It should be emphasized that IAM-TSP provides an approximation solution for the TSP, and it does not ensure finding the optimal solution, because finding the best position for each city alone does not guarantee finding the optimal solution, which necessitates the involvement of all cities at the same time.

In order to improve the IAM-TSP performance, and because involving all cities makes the problem NP-hard, we opt for involving a constant number of local cities (k), and calculate all the permutation sequences starting from the first city in the output Route until the k city, finding the best solution and updating the Rout during this process, after which the algorithm goes into a loop moving by one city to find the next k permutations until $n - k$, this enhanced version is called IAM-TSP+. Algorithm 2 shows the pseudocode processes of the proposed IAM-TSP+, and Fig. 2 depicts the IAM-TSP+ resultant Route of the ATT48 TSPLIB real-world problem [50] in comparison to its optimal route.

Algorithm 1: The proposed IAM-TSP algorithm

Require: Cities xy (List of cities with x and y coordinates) of size n (number of cities)
Ensure: Cost (Total cost of the route using Euclidean distance), and Route (the best possible sequence of cities)
1: Create an empty list Route to store the order of visited cities.
2: Get the East, North, West, and South cities and add them to the Route list. This is done using the minimum/maximum of x and y coordinates.
3: Initialize a Boolean list Visited of size n (number of cities) and set all elements to false.
4: Set the Visited status of the cities in Route to true (East, North, West, and South).
5: Create an empty list Open to store the cities that have not been visited.
6: Add all cities in Cities xy to Open if their Visited status is false.
7: while Open is not empty do
8: Create an empty list of list Possible Solutions to store the possible solutions (routs).
9: for each city i in Open do
10: Find the best location to insert it into Route and add the new route to possible Solutions.
11: end for
12: Find the best solution Best in Possible Solutions and keep track of the city ID.
13: Update Route by Best.
14: Remove the city ID from Open that satisfies Best.
15: end while
16: Calculate the cost of the Route using the Euclidean distance and store it in
Cost.
17: return Cost, Route

Algorithm 2: The proposed IAM-TSP+ algorithm

Require: Cities xy of size n and k=5 (local permutations)
Ensure: Cost, Route
1: Route=IAM-TSP(Cities xy)
2: for i = 1 to n − k do
3: Create an empty list of list Possible Solutions
4: Possible Solutions= all k local permutations of Route(from i to i + k)
5: Update Cost(Possible Solutions)
6: index = *argmin*(Cost)
7: Route = Possible Solutions[index]
8: end for
9: return Cost(Route), Route

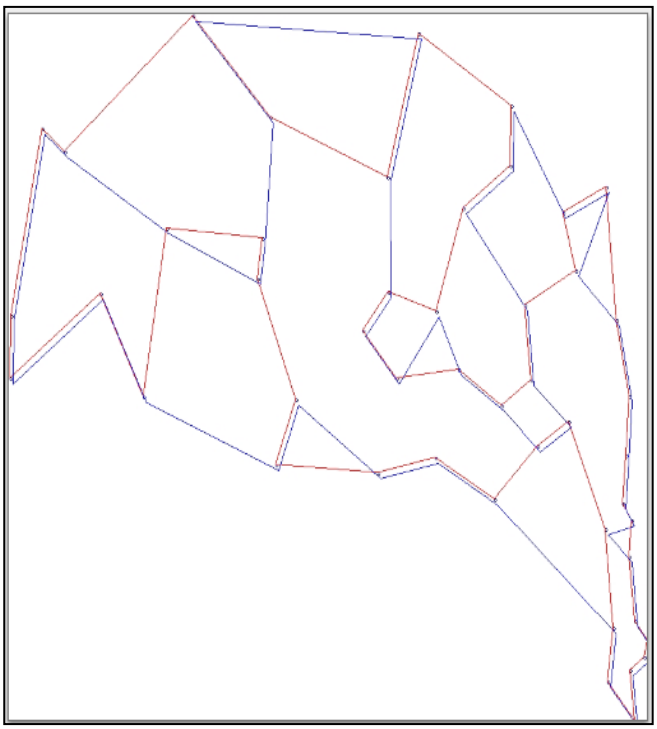

Fig. 2. The IAM-TSP+ resultant route of the ATT48 TSPLIB real-world problem (in blue) n comparison to its optimal route (in red), k = 5, IAM-TSP+ cost = 34877, and the optimal cost = 33523. The optimal route is a little shifted right and down to avoid edge overdrawing.

According to Algorithm 2, the proposed IAM-TSP+ has a time complexity of $O(n^2 + n.k!)$, where n is the number of cities in the Cities xy list. This is because the improved version uses the proposed IAM-TSP as an initial solution, this time is added to the time consumed by sliding the k permutations along the Route, which consumes O(n.k!) in the worst scenario, giving a total time complexity of $O(n^2 + n.k!)$. Since k is a constant number, the time complexity of O(n.k!) is often greater than that of $O(n^2)$. O(n.k!) may still be more efficient than $O(n^2)$ for small values of k, but as (k) grows higher, the growth rate of O(n.k!) quickly exceeds that of $O(n^2)$. Therefore, the time complexity of IAM-TSP+ can be asymptotically approximated to O(n.k!). The space complexity of the IAM-TSP+ is similar to that of the proposed IAM-TSP, with the exception of the space required by the possible Solutions matrix, which contains k! rows of routes, each of which hosts n cities, resulting in a total space complexity of $O(n^2 + n.k!)$, which can be asymptotically approximated to O(n.k!). This is a problem for machines that have limited memory resources, particularly when k is large, and therefore, k needs to be decided based on the available memory resources.

IV. Experimental Setup and Results

To verify the quality and effectiveness of the proposed methods for solving TSP, IAM-TSP and IAM-TSP+ were applied to nine TSP instances, each with a known optimal solution. Those TSPs are from the TSPLIB [50], which has vertices between 40 and 500, namely: a280, att48, berlin52, KroA100, ch150, ch130, pr76, lin105, and pcb442. We chose these specific instances to facilitate comparison to other methods that have been repeatedly used in many studies.

We compare the performance of the proposed methods to other related methods that proposed in the recent years; these include:

- NN: Nearest Neighbor algorithm
- FI: Farthest Insertion algorithm
- CH: Concave hull with Heuristic.
- CNH: Concave hull No Heuristic
- NI: Nearest Insertion algorithm.
- CI: Cheapest Insertion algorithm
- AI: Arbitrary Insertion algorithm
- RNN: Repetitive Nearest Neighbor algorithm
- 2-Opt: 2-Opt algorithm

It deserves to be noted that the aforementioned methods were not developed for this work; rather, we directly reference their results on each standard TSP as stated by [33], where all parameters used for each method can be obtained. It is also worth noting that the majority of these methods' results were obtained by repeating each method a number of times and reporting the average performance. However, we do not need to do the same for the proposed methods because each produces the same result on a specific TSP regardless of how many times the method is run.

We performed simulation experiments using C# of Microsoft visual studio 2022. The hardware and software specifications of the system are as follows:

11th Gen Intel(R) Core (TM) i7-1165G7 @ 2.80GHz, 8.00 GB RAM, and Windows 11 Pro, 64-bit operating system.

The results of the proposed methods to the other compared methods are shown in Table I, in which the optimal tour length (cost) is recorded as reported in the TSPLIB standard library. As it can be seen from Table I, The proposed IAM-TSP+ outperforms the IAM-TSP on all TSPs, which is to be expected given that the latter is an input to the former and the former employs nine local permutations along the resultant route, giving it more chances to identify better solutions.

TABLE I. PERFORMANCE COMPARISON OF THE PROPOSED METHODS TO NINE RELATED METHODS ON DIFFERENT TSP INSTANCES

Instance	Optimal	IAM-TSP	IAM-TSP+	CH	CNH	FI	NI	CI	AI	NN	RNN	2OPT
a280	2579	3051.0804	2978.079155	3335.9	3448	2953	3072.7	3110.7	2995.7	3369.7	3037.7	3018.1
att48	33523	35595.13404	34877.093	35618	35811.2	35267	37893.3	36391.9	35723.1	42112.7	39237	37458.4
berlin52	7542	8497.33404	8031.31761	9013.7	9013.7	8175	9097.7	9007.3	8346.4	9265.4	8182.2	8368.6
ch150	6528	7367.615815	7167.696739	7176.5	7309.6	7148	8066.9	7988.7	7228.8	7734.2	7078.4	7379.4
ch130	6110	6664.37646	6584.106521	7038	7038.8	6855	7381.6	7164.9	6625.8	7747.4	7198.7	6755.5
kroA100	21282	23375.94197	22212.77837	22899.1	23310.1	22874	25957	25073.5	23270.7	27084.2	24699	24401.9
pr76	108159	115547.4825	113155.4313	115790	118877	117173	130029	126837	116098	145227	130921	118838.3
lin105	14379	16285.06763	15948.15549	15596.1	15730.7	15331	18287.9	17327.6	15802.6	18646	16939	16322.2
pcb442	50778	57860.0934	57458.28	66961.6	72425	57537.9	60667.9	59493.1	58001.5	64819.2	59975	57354.4

Furthermore, the proposed IAM-TSP+ outperforms not only IAM-TSP, but also, all methods compared on 5 TSPs, namely att48, berlin52, ch130, KroA100, pr76, followed by the FI method, which also performs well.

Aside from the FI, the proposed IAM-TSP outperforms many other methods without the need for additional improvement and achieves solutions closer to optimal in two instances: att48 and pr76. It is interesting to note that both of the proposed methods achieve performance that is very close to the optimal solution in some instances. In order to illustrate the different performances of both of the best performers, IAM-TSP+ and FI, Fig. 3 compares the resultant routes' costs of both methods on several TSPs while Fig. 4, 5, and 6 provide comparisons of the proposed methods to the other nine related methods on small-scale TSP instances. As it can be seen from these figures, the proposed methods achieve better or comparable performance when compared to the other approximation methods that attempt to find the best possible TSP solution.

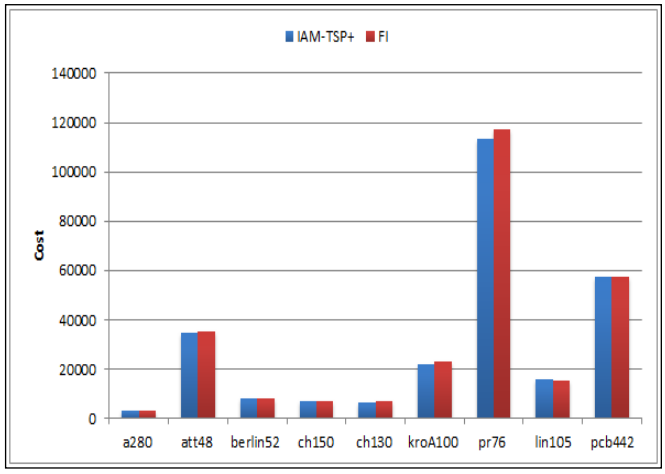

Fig. 3. The routes' cost results of the proposed IAM-TSP+ compared to that of the FI method.

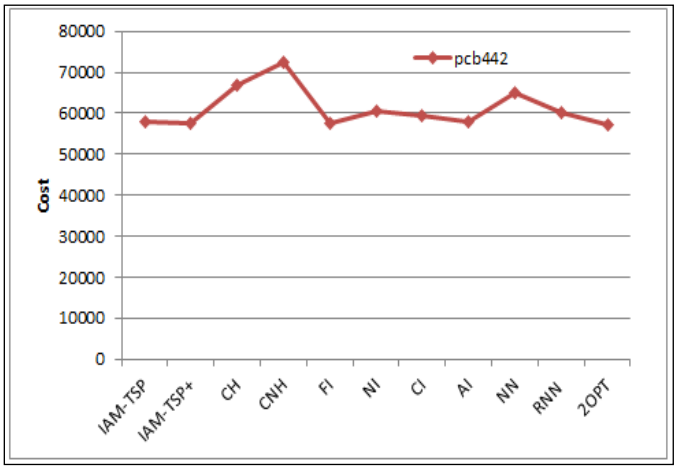

Fig. 4. The routes' cost results of the proposed methods compared to that of the other methods on Att48 TSP.

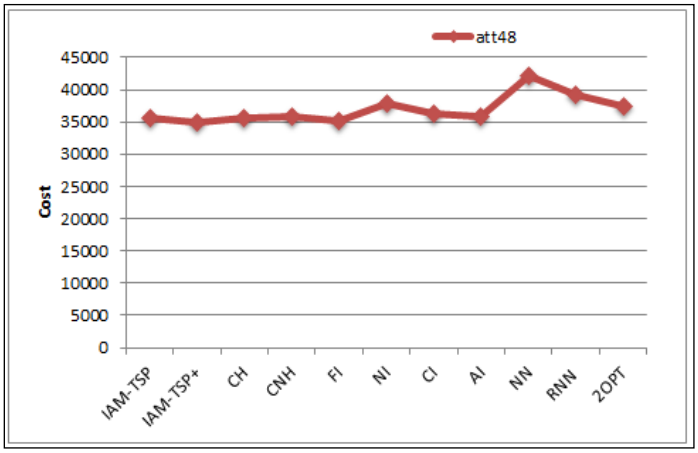

Fig. 5. The routes' cost results of the proposed methods compared to that of the other methods on pcb442 TSP.

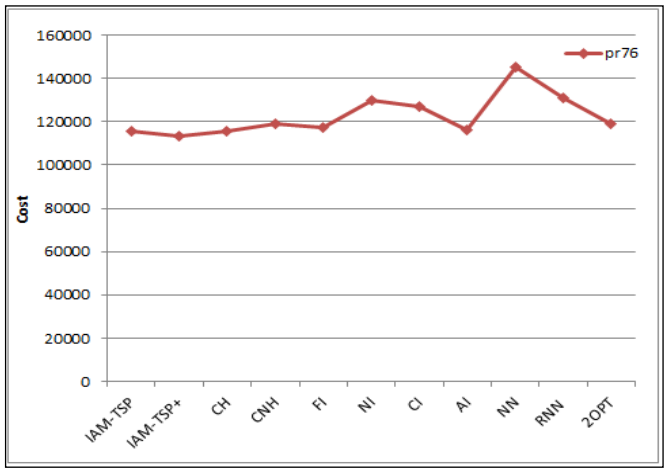

Fig. 6. The routes' cost results of the proposed methods compared to that of the other methods on pr76 TSP.

The approximation ratio, which is the ratio of the method's output to the optimal tour cost, quantifies how much the approximate solution differs from the optimal solution. The approximation ratios of all methods compared are presented in Table II. As it can be seen in the table, the proposed IAM-TSP+ delivers the best overall approximation, with an average of 1.09 overall TSPs, compared to the next competitor (FI), which has an average approximation of 1.10.

The proposed AIM-TSP also performs well, with an average approximation of 1.11, surpassing seven comparable methods. In addition to the approximation ratio, the error rate represents the percentage difference between the solution's fitness value and the known optimal solution, and it is determined as follows [51]:

$$\text{Error Rate} = \frac{solution - optimal\ solution}{optimal\ solution} * 100\% \quad (1)$$

The comparison results for the error rates are presented in Table III. As it can be seen in Table III, the proposed IAM-TSP+ delivers the minimum overall error rate, with an average of 8.51% overall TSPs, compared to the next competitor (FI), which has an average error rate of 9.50%. The proposed AIM-TSP also performs well, with an average error rate of 11.44%, surpassing seven comparable methods. Considering the costs, approximation ratios, and error rates of the resulting Routes, the proposed methods, specifically the IAM-TSP+, outperform all related methods aimed at solving the TSP in general.

TABLE II. COMPARISONS OF THE APPROXIMATION RATIOS OF THE PROPOSED METHODS TO THE OTHER RELATED METHODS

Instance	Optimal	IAM-TSP	IAM-TSP+	CH	CNH	FI	NI	CI	AI	NN	RNN	2OPT
a280	2579	1.183047848	1.15474182	1.2934858	1.33695	1.14502	1.19143	1.20617	1.16157	1.30659	1.1779	1.17026
att48	33523	1.061812309	1.040392954	1.0624944	1.06826	1.05202	1.13037	1.08558	1.06563	1.25623	1.1704	1.117394
berlin52	7542	1.126668528	1.064879025	1.1951339	1.19513	1.08393	1.20627	1.19429	1.10666	1.22851	1.0849	1.1096
ch150	6528	1.128617619	1.09799276	1.0993413	1.11973	1.09498	1.23574	1.22376	1.10735	1.18477	1.0843	1.130423
ch130	6110	1.090732645	1.077595175	1.1518822	1.15201	1.12193	1.20812	1.17265	1.08442	1.26799	1.1782	1.105646
kroA100	21282	1.098390281	1.043735475	1.0759844	1.0953	1.0748	1.21967	1.17816	1.09345	1.27263	1.1605	1.146598
pr76	108159	1.068311306	1.046195243	1.0705535	1.09909	1.08334	1.2022	1.17269	1.0734	1.34271	1.2104	1.098737
lin105	14379	1.132559123	1.109128277	1.0846443	1.09401	1.06621	1.27185	1.20506	1.09901	1.29675	1.1781	1.135142
pcb442	50778	1.139471689	1.131558549	1.3187128	1.42631	1.13313	1.19477	1.17163	1.14226	1.27652	1.1811	1.129513

TABLE III. ERROR RATE COMPARISONS

Instance	Optimal	IAM-TSP	IAM-TSP+	CH	CNH	FI	NI	CI	AI	NN	RNN	2OPT
a280	2579	18.3047848	15.47418203	29.348585	33.6952	14.5017	19.1431	20.6165	16.1574	30.6592	17.786	17.02598
att48	33523	6.181230913	4.039295424	6.2494407	6.82576	5.2024	13.0367	8.558	6.56296	25.6233	17.045	11.7394
berlin52	7542	12.66685282	6.487902542	19.513392	19.5134	8.393	20.6272	19.4285	10.6656	22.8507	8.4885	10.95996
ch150	6528	12.86176188	9.799276026	9.9341299	11.973	9.49755	23.5738	22.3759	10.7353	18.4773	8.4314	13.04228
ch130	6110	9.073264485	7.759517522	15.188216	15.2013	12.1931	20.8118	17.2651	8.4419	26.7987	17.818	10.56465
kroA100	21282	9.839028145	4.37354747	7.59844	9.52965	7.4805	21.9669	17.8155	9.34452	27.2634	16.053	14.65981
pr76	108159	6.831130592	4.619524286	7.0553537	9.90921	8.33403	20.2198	17.2686	7.3404	34.2715	21.045	9.873704
lin105	14379	13.2559123	10.9128277	8.4644273	9.40051	6.62077	27.1848	20.5063	9.90055	29.6752	17.807	13.51415
pcb442	50778	13.94716885	13.1558549	31.871283	42.6307	13.3127	19.4767	17.1631	14.2256	27.6521	18.111	12.95128

V. CONCLUSION

In this paper, we presented two geometric-based approximation greedy algorithms for solving the classical TSP, one of which we call IAM-TSP which is proposed based on locating four extreme nodes/cities and then iterating to determine the best potential positions of each node/city. The other method is known as IAM-TSP+, and it is simply an improved version of the first one, with employing local constant permutations to improve the first method's result.

The experimental results of the proposed methods on nine TSP instances show that, when compared to nine recent related methods, the proposed methods (particularly the IAM-TSP+) provide promising solutions for the classical TSP. This is seen in the resulting routes' cost effectiveness, approximation ratios, and error rates. The enhanced performance (of the IAM-TSP+) is due to the selection of the best local solution, which was accomplished by exploring all k-permutations and considering the best local solution after obtaining the initial solution from the pure IAM-TSP, where k = 5 in all experiments.

The proposed method's limitations include the time and space complexity, which is rather high for the proposed IAM-TSP+, making the evaluation of the proposed methods on large TSP instances particularly difficult on restricted resource machines. As a result, parallel or distributed computation may facilitate the evaluation of proposed methods. In addition to employing the output route as an initial seed for the genetic algorithm [52, 53]. Our future research will focus on such issues.

REFERENCES

[1] M. D. A. C. Hasibuan, et al., Pencarian rute terbaik pada travelling salesman problem (tsp) menggunakan algoritma genetika pada dinas kebersihandan pertamanan kota pekanbaru, SATIN-Sains dan Teknologi Informasi1 (1) (2015) 35–46.

[2] S. Arora, The approximability of np-hard problems, in: Proceedings of the thirtieth annual ACM symposium on Theory of computing, 1998, pp.337–348.

[3] E. Duman, I. Or, Precedence constrained tsp arising in printed circuit board assembly, International Journal of Production Research 42 (1) (2004) 67–78.

[4] F. Su, L. Kong, H. Wang, Z. Wen, Modeling and application for rolling scheduling problem based on tsp, Applied Mathematics and Computation 407 (2021) 126333.

[5] S. Hannenhalli, E. Hubbell, R. Lipshutz, P. A. Pevzner, Combinatorial383 algorithms for design of dna arrays, Chip Technology (2002) 1–19.

[6] S.-Y. Shin, I.-H. Lee, D. Kim, B.-T. Zhang, Multiobjective evolutionary optimization of dna sequences for reliable dna computing, IEEE transactions on evolutionary computation 9 (2) (2005) 143–158.

[7] J. Ahn, E. Choi, D. Lee, Application of routing problems to space exploration missions, in: AIAA SCITECH 2023 Forum, 2023, p. 1966.

[8] D. Ying, Competition decision for bottleneck traveling salesman problem based on big data mining algorithm with multi-segment support, in: 2018 3rd International Conference on Smart City and Systems Engineering (IC-SCSE), IEEE, 2018, pp. 725–729.

[9] B. Jose, T. R. Ramanan, S. M. Kumar, Big data provenance and analytics in telecom contact centers, in: TENCON 2017-2017 IEEE Region 10 Conference, IEEE, 2017, pp. 1573–1578.

[10] A. B. Hassanat, Furthest-pair-based decision trees: Experimental results on big data classification, Information 9 (11) (2018) 284.

[11] S. Mnasri, N. Nasri, T. Val, An overview of the deployment paradigms in the wireless sensor networks, Performance Evaluation and Modeling in Wireless Networks (PEMWN 2014)

[12] A. Abadleh, E. Al-Hawari, E. Alkafaween, H. Al-Sawalqah, Step detection algorithm for accurate distance estimation using dynamic step length, in:2017 18th IEEE International Conference on Mobile Data Management (MDM), IEEE, 2017, pp. 324–327.

[13] S. Tlili, S. Mnasri, T. Val, A multi-objective gray wolf algorithm for routing in iot collection networks with real experiments, in: 2021 National Com-407 puting Colleges Conference (NCCC), IEEE, 2021, pp. 1–5.

[14] A. Mars, A. Abadleh, W. Adi, Operator and manufacturer independent d2d private link for future 5g networks, in: IEEE INFOCOM 2019-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), IEEE, 2019, pp. 1–6.

[15] A. Aljaafreh, K. Alawasa, S. Alja'afreh, A. Abadleh, Fuzzy inference system for speed bumps detection using smart phone accelerometer sensor, Journal of Telecommunication, Electronic and Computer Engineering 9.

[16] A. Abadleh, B. M. Al-Mahadeen, R. M. AlNaimat, O. Lasassmeh, Noise segmentation for step detection and distance estimation using smartphone sensor data, Wireless Networks 27. doi:10.1007/s11276-021-02588-0.

[17] N. Ghatasheh, H. Faris, R. Abukhurma, P. A. Castillo, N. Al-Madi, A. M. Mora, A. M. Al-Zoubi, A. Hassanat, Cost-sensitive ensemble methods for bankruptcy prediction in a highly imbalanced data distribution: A real case from the spanish market, Progress in Artificial Intelligence 9 (2020) 361–375.

[18] G. A. Altarawneh, A. B. Hassanat, A. S. Tarawneh, A. Abadleh, M. Alrashidi, M. Alghamdi, Stock price forecasting for jordan insurance companies amid the covid-19 pandemic utilizing off-the-shelf technical analysis methods, Economies 10 (2) (2022) 43.

[19] A. B. Hassanat, V. S. Prasath, M. Al-kasassbeh, A. S. Tarawneh, A. J.Al-shamailh, Magnetic energy-based feature extraction for low-quality fin-429 gerprint images, Signal, Image and Video Processing 12 (2018) 1471–1478.

[20] A. S. Tarawneh, C. Celik, A. B. Hassanat, D. Chetverikov, Detailed investigation of deep features with sparse representation and dimensionality reduction in cbir: A comparative study, Intelligent Data Analysis 24 (1) (2020) 47–68.

[21] E. Hamadaqa, A. Abadleh, A. Mars, W. Adi, Highly secured implantable medical devices, in: 2018 International Conference on Innovations in Information Technology (IIT), IEEE, 2018, pp. 7–12.

[22] A. S. Tarawneh, A. B. Hassanat, D. Chetverikov, I. Lendak, C. Verma, Invoice classification using deep features and machine learning techniques, in:2019 IEEE Jordan International Joint Conference on Electrical Engineering and Information Technology (JEEIT), IEEE, 2019, pp. 855–859.

[23] A. B. Hassanat, On identifying terrorists using their victory signs, Data Science Journal 17.

[24] W. J. Cook, In pursuit of the traveling salesman, in: In Pursuit of the Traveling Salesman, Princeton University Press, 2011.

[25] A. B. Hassanat, E. Alkafaween, On enhancing genetic algorithms using new446 crossovers, International Journal of Computer Applications in Technology 55 (3) (2017) 202–212.

[26] A. Hassanat, K. Almohammadi, E. Alkafaween, E. Abunawas, A. Hammouri, V. S. Prasath, Choosing mutation and crossover ratios for genetic algorithms—a review with a new dynamic approach, Information 10 (12) (2019) 390.

[27] E. Alkafaween, A. B. Hassanat, Improving tsp solutions using ga with a new hybrid mutation based on knowledge and randomness, Communications Scientific letters of the University of Zilina 22 (3) (2020) 128–139.

[28] P. Hart, The condensed nearest neighbor rule (corresp.), IEEE transactions on information theory 14 (3) (1968) 515–516. [29] G. A. Croes, A method for solving traveling-salesman problems, Operations research 6 (6) (1958) 791–812.

[29] G. A. Croes, A method for solving traveling-salesman problems, Operations research 6 (6) (1958) 791–812.

[30] D. J. Rosenkrantz, R. E. Stearns, P. M. Lewis, Approximate algorithms for the traveling salesperson problem, in: 15th Annual Symposium on Switching and Automata Theory (swat 1974), IEEE, 1974, pp. 33–42.

[31] J. Brest, J. Zerovnik, An approximation algorithm for the asymmetric traveling salesman problem, Ricerca operativa 28 (1999) 59–67.

[32] T. Wainwright, Solving problems involving hamilton circuits, Journal of Mathematics and Science: Collaborative Explorations 1 (1) (1997) 83–90.

[33] K. Ihsan Kilic, L. Mostarda, Novel concave hull-based heuristic algorithm for tsp, in: Operations Research Forum, Vol. 3, Springer, 2022, p. 25.

[34] K. Yang, X. You, S. Liu, H. Pan, A novel ant colony optimization based on game for traveling salesman problem, Applied Intelligence 50 (2020) 4529–4542.

[35] M. Mosayebi, M. Sodhi, T. A. Wettergren, The traveling salesman problem with job-times (tspj), Computers & Operations Research 129 (2021) 105226.

[36] M. Mosayebi, The variants of traveling salesman problem with job-times (tspj).

[37] P. He, J.-K. Hao, Hybrid search with neighborhood reduction for the multiple traveling salesman problem, Computers & Operations Research 142 (2022) 105726.

[38] D. A. Aliyev, C. L. Zirbel, Seriation using tree-penalized path length, European Journal of Operational Research 305 (2) (2023) 617–629.

[39] A. Maya-L´opez, A. Mart´ınez-Ballest´e, F. Casino, A compression strategy for an efficient tsp-based microaggregation, Expert Systems with Applications 213 (2023) 118980.

[40] D. Rodr´ıguez, J. M. Cruz-Duarte, J. C. Ortiz-Bayliss, I. Amaya, Asequence-based hyper-heuristic for traveling thieves, Applied Sciences 13 (1) (2023) 56.

[41] V. Cacchiani, C. Contreras-Bolton, L. M. Escobar-Falc´on, P. Toth, Amatheuristic algorithm for the pollution and energy minimization traveling salesman problems, International Transactions in Operational Research 30 (2) (2023) 655–687.

[42] H. Liu, A. Lee, W. Lee, P. Guo, Daaco: adaptive dynamic quantity of ant aco algorithm to solve the traveling salesman problem, Complex & Intelligent Systems (2023) 1–14.

[43] S. S. Harahap, P. Sihombing, M. Zarlis, Combination of ant colony tabu search algorithm with firefly tabu search algorithm (acts-fats) in solving the traveling salesman problem (tsp), Sinkron: jurnal dan penelitian Teknik informatika 8 (1) (2023) 212–221.

[44] S. Dhouib, A new column-row method for traveling salesman problem: the dhouib-matrix-tsp1, International Journal of Recent Engineering Science 8 (1) (2021) 6–10.

[45] E. Osaba, X.-S. Yang, J. Del Ser, Traveling salesman problem: a perspective review of recent research and new results with bio-inspired metaheuristics, Nature-Inspired Computation and Swarm Intelligence (2020) 135–164.

[46] Cengiz, C. Yilmaz, H. T. Kahraman, C¸. Sui¸cmez, Effects of variable uav speed on optimization of travelling salesman problem with drone (tsp-d), in: Smart Applications with Advanced Machine Learning and Human-Centred Problem Design, Springer, 2023, pp. 295–305.

[47] N. Garg, M. K. Kakkar, G. Gupta, J. Singla, A performance evaluation of genetic algorithm and simulated annealing for the solution of tsp with profit using python, in: Emerging Technologies in Data Mining and Information Security: Proceedings of IEMIS 2022, Volume 3, Springer, 2022, pp. 13–26.

[48] M. Das, A. Roy, S. Maity, S. Kar, A quantum-inspired ant colony optimization for solving a sustainable four-dimensional traveling salesman problem under type-2 fuzzy variable, Advanced Engineering Informatics 55 (2023) 101816.

[49] ´E. D. Taillard, Design of Heuristic Algorithms for Hard Optimization: With Python Codes for the Travelling Salesman Problem, Springer Nature, 2023.

[50] G. Reinelt, Tsplib—a traveling salesman problem library, ORSA journal on computing 3 (4) (1991) 376–384.

[51] S. S. Ray, S. Bandyopadhyay, S. K. Pal, Genetic operators for combinatorial optimization in tsp and microarray gene ordering, Applied intelligence 26 (2007) 183–195.

[52] E. Alkafaween, A. B. Hassanat, S. Tarawneh, Improving initial population for genetic algorithm using the multi linear regression based technique (mlrbt), Communications-Scientific letters of the University of Zilina 23 (1) (2021) E1–E10.

[53] A. B. Hassanat, V. S. Prasath, M. A. Abbadi, S. A. Abu-Qdari, H. Faris, An improved genetic algorithm with a new initialization mechanism based on regression techniques, Information 9 (7) (2018) 167.