International Journal of Psychosocial Rehabilitation, Vol. 24, Issue 05, 2020
ISSN: 1475-7192

Traveling Salesman Problem Methods of Solution Survey

[1]Warif B. Yahia, [2]Ghassan E. Arif, [3]Mohammed W. Al-Neama, [4]Ali Hasan Ali

Abstract

In this paper, a full review of the Travelling Salesman problem (TSP) is given. In general, TSP is considered an important area of research in the field of the applied mathematics. The problem has started in the 19th century by the well-known mathematician ((Hamilton)). The idea of TSP is to find the shortest Hamiltonian cycle that covers all the nodes in a graph. Later, researchers started to work on this problem using a variety of techniques to obtain an optimal solution. In fact, solutions of TSP are divided into two main types. The exact solutions that do not require a long time to be processed and the heuristic solutions that give approximate results. The main idea of this paper is to review a collection of successful methods that deal with TSP. In addition, this paper contains some examples with different databases size and comparison of multiple methods. Furthermore, a mathematical model of this problem with complexity analysis are given.

Keywords: *Travelling Salesman problem (TSP), applied mathematics*

I. Introduction

The Traveling Salesman Problem (TSP) is one of the most important problems in the field of computational and applied mathematics regarding its applications in real-life problems ,as well as, it has a relationship with many other fields [44]. In addition, it was classified as one of the hardest problems because it falls within the complexity class of the NP-complete. The complexity of this problem has been inspiring many researchers to find and modify new ways to reduce the complexity of this problem and trying to reach an optimal solution efficiently.

Basically, the TSP as shown in figure (1) can be defined and explained as a set of cities that the salesman is wanted to visit in a shortest possible way in order to reduce the cost of the trip and then return to the city from which he started.

TSP in the computational complexity theory has been classified as an NP-complete problem, which means if there are n cities in a complete graph then there are $(n-1)!\,/\,2$ possible different tours for the salesman. If n is a large number, then the classical methods take a huge amount of time to find

[1] *College of Education for Pure Sciences, Tikrit University*
[2] *College of Education for Pure Sciences, Tikrit University*
[3] *Education College for Girls, Mosul University*
[4] *College of Education for Pure Sciences, University of Basrah*

Received: 02 March 2020 | Revised: 21 April 2020 | Accepted:05 May 2020

the optimal solution. Therefore, researchers in the field of artificial intelligence always seek to find new ways to solve this problem and try to reduce its complexity.

Scientists have simulated the behavior of many living creatures that have less intelligence than human beings such as ants, bees, birds, and many other animals that live in swarms. This simulation helped the researchers to solve many difficult problems, including the traveling salesman problem.

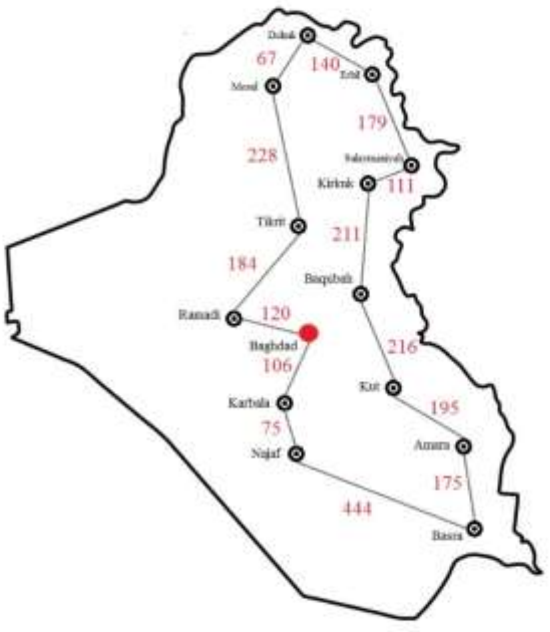

Figure 1: A tour of traveling salesman in 14 cities in iraq.

II. TSP and Graph Theory

In graph theory, the problem is represented by a graph consisting of a number of nodes that represent the cities and these nodes are connected to each other by edges. Each edge represents the road between any two cities. All edges in this graph have a weight that represents a cost, distance, or time. This type of graph is called a complete weighted graph. The primary goal is to find a Hamiltonian cycle on these cities with the lowest possible distance [26].

III. Mathematical Model of Traveling Salesman Problem

If $G = (V, E)$ is a weighted complete graph where V is the set of nods such that $|V| = n$ and E is the set of edges. Each node in the graph is connected with $n - 1$ edges. The weight of each edge is given by D, where D is $n \times n$ distance matrix. The mathematical formulation of the TSP is given by the following objective function:

$$\min(h) = \sum_{i=1}^{n} \sum_{j=1}^{n} d_{ij} \mathcal{L}_{ij} \dots \dots \dots (1)$$

where, $i, j \in V$, $d_{ij} \in D$ is the distance from i to j.

subject to $$d_{ij} = d_{ji} = 0 \; if \; i = j \dots\dots\dots (2)$$

$$\sum_{i=1}^{n} \mathcal{L}_{ij} = 1, \;\; j = 1, \dots, n \;\; \dots\dots\dots (3)$$

$$\sum_{j=1}^{k} \mathcal{L}_{ij} = 1, \;\; i = 1, \dots, n \;\; \dots\dots\dots (4)$$

Otherwise $\mathcal{L}_{ij} = 0 \;\; \dots\dots\dots (5)$

The tour in TSP is Hamiltonian cycle $\dots\dots\dots (6)$

$\mathcal{L}$ is the decision variable determines whether or not the salesman decides to move. $\mathcal{L}_{ij}$ is equal to 1 if the salesman is traveling from the city i to j and $\mathcal{L}_{ij}$ is equal to 0 if the salesman is traveling from the city i to another city that is not j.

The constrains (3) and (4) are known conditions of assignment problem [19] which means that the salesman must visit one city after he visits the city i, and the salesman when he is in j must come from only one city respectively [43]. The constrain (2) ensures that there is no loop in the tour, also (6) confirms that the salesman returned to the first city when he reached the last city.

IV. The Classifications of TSP

According to the applications of this problem in life. TSP has been classified into three main types:

1- Symmetric TSP: The TSP is called symmetric if the distance from the city i to j is equal to the distance from the city j to i in the distance matrix D [16]. This means $d_{ij} = d_{ji}$. This is the normal case of a traveling salesman problem in general and is denoted by TSP. The study of this thesis is concerned with this case only.

2- Asymmetric TSP: The TSP is called asymmetric if the distance from the city i to j is not equal to the return distance [9]. This means $d_{ij} \neq d_{ji}$ in D or if the graph is not complete (not all cities are connected to each other's). In this case, the traveling salesman problem is denoted by ATSP.

3- Dynamic TSP: The TSP is called dynamic if the nods in the graph are moving. As if these points are customers whose positions are changing for the salesman [27]. In this case, the traveling salesman problem is denoted by DTSP.

V. Best Tour

The tour in TSP is represented by its Hamiltonian cycle which is given by a sequence of nodes that are connected by edges and the distance of the tour is equal to the sum of all weights on the edges in the cycle.

The graph in TSP is two-dimensional and the distance can be found by using the Euclidean distance [19], where the nodes i and j are points on a coordinate plane:

$$d_{ij} = \sqrt[2]{\left(x_j - x_i\right)^2 + (y_j - y_i)^2}$$

Where the distance in the graph satisfied the triangle inequality. If the graph does not satisfy this condition, then no Hamiltonian cyclic can be found. Also, every graph that contains 4 nodes and more has at least one Hamiltonian cyclic that not cross itself [5] and this condition is very important to find the best tour as shown in figure (2) the distance of the tour which is in bold line in (A) is equal to 16 where the distance of the tour in (B) is equal to 14, and this happened because the tour in (A) is cross itself, thus it must find a method of solution that can reduce the occurrence of this situation to get a best tour.

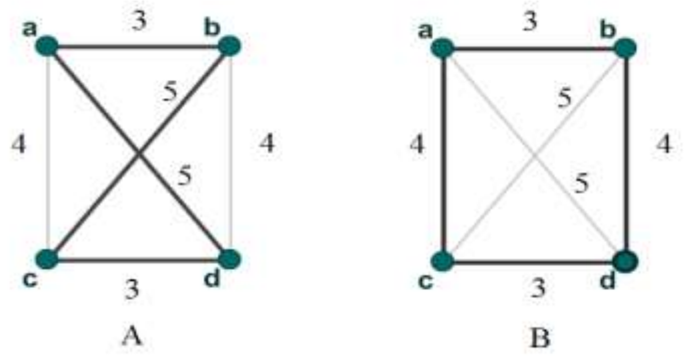

Fig. 2: The difference in the distance between a Hamiltonian cyclic that crosses itself and one that dosn't crosses itself in
the same weight complete graph.

VI. TSP Complexity

If the salesman positioned in a graph contains n cities then he must visit $(n-1)$ cities, that means there are $(n-1)!$ possible Hamiltonian cycles.

6.1 Theorem [5]: Let H be a set of all possible tour and $h_1, h_2 \in H$ are two tours in which the nodes are visited in reverse order, if D is symmetric distance matrix then the distance of $h_1 = h_2$.

From theorem (2.6.1), the salesman has $(n-1)!\ /\ 2$ possible tours. To find the exact solution, all possible tours must be tested, and the tour that has a least-cost must be chosen. Thus, the Big O-notation is equal to $O((n-1)!/2)$ and this complexity is very big. Moreover, TSP is considered as an NP-complete problem and the following theorem will illustrate that.

6.2 Theorem: TSP is an NP-complete problem

Proof: Let $G = (V, E)$ is weight complete graph with n nods, and TSP is defined on G. To show that $TSP \in NP^c$ then by definition (1.6.3.4). It must to show that $TSP \in NP \; and \; TSP \; is \; NP - hard$. If we have at least one tour on G. Then the total cost T of this tour can be calculated by the following formula

$$T = \sum_{i=1}^{n} x_i, \text{where } x_i \text{ is the cost on each edge}$$

T can be solved by NDA_{poly}. Then by definition (1.6.3.2) $TSP \in NP$.

Now, to show that TSP is NP-hard problem. Let h be a Hamiltonian cycle defined in G. Since h is NP^c, then h is NP. It is known that $TSP = h + T$, that means $TSP \geq h$. Then by definition (1.6.3.3) TSP is NP^h.

Thus, TSP is NP^c. ■

Since the complexity of TSP is factorial, it needs an efficient method in order to find the best tour.

VII. TSP Methods of Solution

Many algorithms have been designed to solve TSP. There are differences in these algorithms in terms of time and space that are taken to reach the solution. These methods are divided into two main approach:

- Exact algorithms: in NP-class the exact algorithms are the methods that designed to gain an exact solution. Where the Big O-notation of it is factorial. Figure (1.13) shows that this running time is long whenever the value of n is big. Therefore, reaching the exact solution will be very difficult.
- Heuristic algorithms: in any optimization problem heuristic algorithms are algorithms that equipped with techniques organized to solve a specific problem and try to reach the optimal solution or fall near of optimal solution. After that, the heuristic algorithms were developed and augmented with more than one strategy during implementation, and it were circulated to solve many problems and called a meta-heuristic algorithms. The complexity of these algorithms is less than the exact algorithms. Therefore, it gives solutions in less time and space. The heuristic algorithms, in general, are usually used in cases where the approximate solution is sufficient for the problem in particular when reaching the exact solution is very difficult. But the aspiration is always to reach the exact solution.

In each approach, many methods were presented in the previous years in exact approach, in (1962), Bellman proposed a dynamic programming method as an exact method to solve TSP [6]. As well, in (1963), a new method was designed to solve TSP called Branch and Bound algorithm (B&B) and it is proposed by Little *et al.* also as an exact algorithm [24].

In the heuristic approach. In (1971), Krolak, Felts, and Marble introduced a new method to solve TSP. The method was called Man-Machine Approach. The method was applied using a UNIVersal Automatic Computer

(UNIVAC). They used several datasets to explain the method where the number of cities was less than 200 cities [22].

A second goal of the research was to compare the time of resolving TSP using a UNIVAC computer with other computers older than it, where the results showed a significant improvement at the time of implementation and CPU processing [22].

In (1985), Grefenstette *et al*. applied the Genetic Algorithm (GA) on TSP. Where this algorithm relies on biologically inspired factors such as hereditary mutations. GA was applied to many known datasets of TSP, and they found that the algorithm generates very accurate and efficient solutions [15].

Tabu Search (TS) is one of the important methods that solve TSP. This method was introduced by Glover in (1989). Glover obtained good results when he applied on known datasets of TSP containing 42 and 75 cities [14].

In (1992), Colorni, Dorigo, and Maniezzo used the Ant Colony Optimization (ACO) to solve TSP. ACO was applied to the known dataset (Oliver30) that contains 30 cities. ACO was compared to GA in this dataset. The results indicated that the two algorithms reached the same tour length (424,635). That means ACO is efficient and can be improved to get better results [11].

In (2003), Wang *et al*. used the particle swarm optimization, which was presented for the first time in (1995), to solve combinatorial optimization problems. The method was applied to TSP and called it the Particle Swarm Optimization (PSO). PSO was programmed with C ++ and applied to one dataset containing 14 cities. The results showed that PSO reached the optimal solution in this dataset. Also, it was very fast in convergence towards a solution and the space used in the solution was very small [41].

In (2007), Bakhouya and Gaber presented a suggested method for solving TSP using Immune Inspired-based Optimization Algorithm (IIOA). IIOA was programmed using Java and it was applied to many datasets, with number of cities reaching 1000. The results were compared with the ACO in terms of the time taken by the CPU to implement and also the length of the tour. The results which obtained showed that IIOA overcame the ACO in terms of time and length of the tour in all datasets [4].

In (2010), an algorithm was derived from the geographical distributions of biological species and applied to solve TSP. It was proposed by Mo and Xu the algorithm was called TSP Biogeography Migration Algorithm (TSPBMA). The algorithm was applied to well-known datasets of TSP (oliver30, eil50, eil75, and kroA130). The results were compared with five other algorithms derived from the nature. It was significantly outperformed four of them. However, it was unable to find any improvement in front of the ant colony's algorithm, especially in the matter of convergence to the solution [28].

In (2011), Cheng *et al*. Introduced a new algorithm derived from nature. Cheng was studied the behavior of the cockroach in terms of living in a swarm, how to search for food, and equality between individuals. This algorithm was called Cockroach Swarm Optimization algorithm (CSO) and applied to TSP [10].

The paper covers only one known dataset, which is (Oliver30) that contains 30 cities. The comparison was made with ACO and PSO in terms of cost only. The results showed that there was no improvement when CSO compared to ACO. However, in front of PSO, it was significantly decreased the cost of the tour, where the error percentage was 0.05%, while in the PSO 0.28% in the same dataset [10].

In (2011), Marinakis, Marinaki, and Dounias introduced a proposed algorithm to solve TSP, using Honey Bees Mating Optimization (HBMO) and adapting it to solve the discrete problems using Multiple Phase Neighbourhood Search-Greedy Randomized Adaptive Search Procedure (MPNS-GRASP) [25].

The new algorithm was designed by using Fortran and called it Honey Bees Mating Optimization for TSP (HBMOTSP). HBMOTSP was applied to 74 known datasets of TSP with a very large number of cities reaching 85900 cities. HBMOTSP has proven very high efficiency and has overcome many previous solution methods, where it reached the optimal solution in 63 datasets. On the other hand, the error for the remaining datasets was very low, it estimated at 0.1% [25].

In (2011), Jati developed an important evolutionary algorithm that could deal with discrete spaces. He called it Evolutionary Discrete Firefly Algorithm (EDFA), EDFA was applied to TSP by using seven datasets. The results showed a high accuracy of this algorithm. Where it was able to reach the optimal solution in four datasets. Its results were also compared with the Memetic algorithm and there was a great convergence between the results [20].

In (2011), Karaboga and Gorkemli modified the Artificial Bee Colony algorithm (ABC). In order to implement it on discrete space. A Combinatorial Artificial Bee Colony (CABC) version was applied on TSP, by using two important known datasets of TSP which are KroB150 and KroA200 that contains 150 and 200 cities respectively. Also, CABC compared with eight different versions of the genetic algorithm. CABC achieved little error of 0.9847 and 0.6211 for the two datasets, respectively [21].

In (2013), the cat swarm optimization, which appeared in (2007), was redesigned to solve TSP by Bouzidi and Essaid Riffi. The method was called Discrete Cat Swarm Optimization (DCSO). DCSO was applied to 16 known datasets. The results showed that DCSO was able in one of the iterations to reach the optimal solution in all datasets. Where the number of implementations was 10. There was a slight error in the remaining implementations. DCSO was not compared to any other method, due to its optimization of all datasets [8].

In (2013), Sur, Sharma, and Shukla presented a suggested method for solving TSP using the Egyptian Vulture Optimization Algorithm (EVOA). It is an algorithm inspired by nature. The results of EVOA was not compared to any other algorithms. EVOA applied to 9 known datasets of TSP. The number of cities in these datasets ranges from 16 to 280 cities. After implementing EVOA 25,000 times for each dataset. EVOA has a different error average by comparing the algorithm solutions with the optimal solution of each dataset. Where the best error average is 4.6 in the dataset that contains 16 cities, and the worst-case is 61.2 with a dataset containing 202 cities [39].

Also in (2014), Verma, Jain, and Chhabra used a new algorithm that simulated the behavior of chemical movement in bacteria and applied it to TSP. This algorithm was called Bacterial Foraging Optimization Algorithm (BFOA). BFOA was applied to many datasets suggested by researchers. Only time has been studied and compared to GA. The study showed a significant improvement at the time of finding the solution of TSP [40].

The behavior that the bat uses in life has been studied. It uses echo to find prey as well to avoid obstacles in the dark. In (2014), Saji, Essaid, and Ahiod conducted a study on this algorithm but with its discrete version and called it Discrete Bat-inspired Algorithm (DBA). They applied DBA to TSP. The results of DBA was compared with nine other algorithms, using five known datasets of TSP. The algorithm has made remarkable progress with 60% of the datasets, especially those that contain less than 76 cities [34].

In (2014), a new suggested method was presented, which inspired by the behavior of coral reefs. Where it lives in colonies and trying to create new areas for growth by stifling other coral colonies. This method was designed to solve TSP by Salcedo-Sanz *et al*. Coral Reefs Optimization algorithm (CRO) was applied to datasets suggested by the researchers, the datasets were divided into two categories, small data category ranges from 15 to 120 cities, and big data category where the range from 200 to 400 cities. The results were compared with the harmonic algorithm and PSO, and measured using the standard division. The algorithm showed very high accuracy and noticeable superiority over the rest of the algorithms [35].

The scientific algorithm was applied to combinatorial optimization problems in (2014) by Felipe, Ferreira, and Goldbarg. They presented research dealing with TSP and applied to the car renter problem. The method was programmed using C++. The scientific algorithm has had several experiments and had been compared to Transgenetic Algorithms. The results showed a great advantage for the scientific algorithm, especially at the time of implementation, where the time was reduced vary significantly [13].

A Water Flow-like Algorithm (WFA) was introduced in (2014), to solve TSP by Srour, Ali, and Hamdan. WFA was designed using (Java language) and applied to 23 known dataset of TSP, with number of cities ranging from 51 to 3975. WFA was compared at first with ACO. The results showed that the proposed algorithm was superior in the average of solutions in 10 implementations with 10,000 iterations. Then it was compared with many algorithms, WFA was very efficient and has high accuracy in solutions [38].

An algorithm simulating lightning was also used. Where lightning was considered as a process of reducing shipments in the clouds. This simulation was applied on TSP in reducing tour cost. The method was introduced in (2015), by El Majdouli and El Imrani. It was called Lightning Inspired Search Algorithm (LISA). LISA has been applied to five known datasets of TSP, with number of cities ranging from 51 to 198 cities. LISA was compared with several algorithms in terms of the cost of the tour only. This paper states that the proposed method has overcome all algorithms [12].

The cuttlefish optimization was also improved to work with TSP in (2015), by Riffi and Bouzidi. The method was called Discrete Cuttlefish Optimization Algorithm (DCOA). DCOA was designed using C++ language and applied to 37 known datasets of TSP. DCOA was implemented ten times on each dataset. The results showed that DCOA reached the optimal solution for all datasets except one where the error average was very small. DCOA was also compared with several other algorithms, where DCOA showed high accuracy in solutions, as well as an efficient implementation time [33].

In (2015), Zhou *et al*. designed a new method using Discrete Invasive Weed Optimization Algorithm (DIWO). DIWO has been tested on 20 known datasets of TSP. DIWO achieved good results in the datasets that include less than 280 cities with an average of error was less than 0.1%. On the other hand, in the datasets that contain a large number of cities, there was an observed error rate. The results of DIWO were compared with several other algorithms using 7 datasets. DIWO outperformed the other algorithms in most datasets [45].

In (2016), Agharghor, Essaid and Chebihi used the natural behavior of animals to hunt on TSP. The method was called Memetic Hunting Search algorithm (MHuS). MHuS has been applied to 10 datasets, where the number of cities in these datasets ranges from 70 to 124. The results showed that the average of ten implementations of MHuS in all datasets was equal to the optimal solution [1].

The Monarch Butterfly Optimization (MBO) that studies butterflies' immigration behavior in America has been improved to work on combinatorial optimization problem in (2016) by Wang *et al*. The method was called Discrete Monarch Butterfly Optimization (DMBO). DMBO was applied to 4 unknown datasets given by researchers containing (50, 100, 150, and 200) cities. The results were compared with 3 other algorithms: DE, ACO, and Biogeography-based optimization. DMBO excelled in finding the best tour in three datasets. It also achieved the best average of solutions in two datasets [42].

In (2017), the Soccer Game Optimization (SGO) was applied to solve TSP by Purnomo *et al*. SGO was compared with three versions of the neural network method using 25 known datasets. SGO was able to reach the optimal solution with only one dataset. Also, SGO excels at reaching the best possible tour with 68% of the datasets. SGO was also compared in terms of implementation time, SGO was outperformed by only one dataset. On the other hand, SGO was compared with four different versions of GA in terms of tour length over 15 datasets, as it reached the best solution in 8 datasets, but it was not superior in time terms to any of the other algorithms [31].

In (2017), the Plant Propagation Algorithm (PPA) -first introduced in 2016- was implemented to solve TSP by Selamoğlu and Salhi. They called it Discrete Plant Propagation Algorithm (Discrete PPA). Discrete PPA was applied to 10 known datasets of TSP with the number of cities between 14 to 101. The results were compared with the GA and Simulated Annealing (SA). The results showed high accuracy in the solution as well as a significant improvement in time, especially in matters that contain less than 22 cities. Also, Discrete PPA compared with four versions of PSO in 4 datasets it was significantly outperformed by the accuracy of the solution. Moreover, in the datasets that contain more than 101 cities, Discrete PPA compared with EDFA and the accuracy of the solutions was exceed 96.32% [36].

The swallow swarm optimization first appeared in (2013) with its continuous version. Then, in (2017), Bouzidi and Essaid improved this algorithm to work in discrete space and called it Discrete Swallow Swarm Optimization (DSSO). DSSO was applied to TSP using C++ language. The researchers used ten known datasets ranging from 51 to 225 cities. DSSO reached the known optimal solution for eight datasets and it was very closed to the optimal in the rest. Also, DSSO was compared to DBA and PSO, and it surpassed them in the accuracy of the solution [7].

In (2018), Hammouri *et al*. developed the Dragonfly Algorithm (DA), one of the swarm intelligence algorithms, to work with combinatorial optimization problems using Java. DA was applied to TSP by using 10 known datasets, the number of cities of these datasets are between 22 and 101. The result of DA was compared in terms of time and cost of the tour with four algorithms (PSO, GA, BH, and ACO). The results showed that the proposed method exceeded all algorithms in terms of time. But it did not achieve any superiority over the cost of the tour [17].

Also in (2018), the clockwise search method was used and modelled with galaxy based search algorithm to design a new method using MATLAB® to solve TSP by Phu-ang. The proposed algorithm has been tested on 4 datasets, the number of cities in these datasets was from 150 to 200. The results compared with two other algorithms. The proposed algorithm showed great superiority and high accuracy, where its mean relative error was 0.114%, which was less than the best algorithm compared to it by 0.351% [30].

The standard Grey Wolf Optimization (GWO) also has been improved to work with combinatorial optimization problem in (2018), by Sopto *et al*. GWO has been applied to TSP, where each wolf represents a tour for the salesman. A Swap Operator(SO) has been added to improves the final solution resulting from each wolf. GWO was implemented on 15 known datasets of TSP, as the number of cities in these datasets ranges from 14 to 100. It was compared with GA and ACO. GWO surpassed them with high accuracy by 10 datasets, which ranges from 14 to 51 cities. On the other hand, ACO has excelled in solving the remaining datasets [37].

In (2018), Hatamlou introduced a new way to solve TSP. This method is inspired by the phenomenon of the black hole and was called the Black Hole algorithm (BH). BA has been applied to ten datasets where the number of cities ranges from 22 to 101. It is compared with three algorithms ACO, PSO, and GA using MATLAB®. The algorithm showed distinctive and more accurate solutions specifically in terms of implementation time [18].

The Spider Monkey Optimization (SMO) was used in (2019), by Ayon *et al.* to solve TSP. SMO was programmed by using MATLAB®. The new method was applied to 15 known datasets, where the number of cities in the datasets was between 14 to 101. SMO was compared with ACO and Velocity Tentative PSO (VTPSO). The results showed that SMO outperformed in the average of implementation in 11 datasets with 10 runs. Also, SMO showed a rapid convergence towards a solution [3].

In (2019), Ahmed, Sadiq, and Abdullah proposed a new algorithm to solve TSP. This algorithm examines the behavior of camels and how it transmitted in the desert environment, as well as the ways of communication between them. The algorithm was called Camels Herd Algorithm (CHA). The paper shows that CHA was able to solve TSP. It applied to 18 TSP known datasets, and It was compared with four other algorithms (ACO, PSO, ABC, and HA). The comparison was in terms of the cost only and nothing was mentioned about the time of implementation. However, no improvement was observed in many datasets. But CHA in all datasets overcame artificial bee colony [2].

VIII. Comparative Study and Conclusion

Since the methods of solving a TSP differ in the way of reaching optimal or approximated solutions, many criteria have to be taken into consideration when deciding which method to use. Some researchers have been produced for surveying, comparing and evaluating TSP methods to address this critical issue and highlight a number of specific strengths and weaknesses of them.

The assessment and the choice of the most convenient methods are subjected to variant metrics. In the following, we summarize the most important metrics that affect the usability and popularity of the application.

• **Availability:** the ease to obtain and use the codes. The code needs to be publicly available and user-friendly so that anyone can apply it and compare its results with others. Table (1) lists the available TSP codes which are published in github https://github.com/ and Mathworks https://www.mathworks.com/.

• **Portability:** The ability to run the program with different OS is very significant as most researchers intend to run it on their PCs. Table (1) in the end of the section shows that the programming languages and programs (C ++, C, C #, Fortran, CUDA, Java, JavaScript, MATLAB®, and Python) are available to most researchers and it can be obtained easily. In addition, all of these programming languages and programs can be run on Microsoft Windows OS which is the most used OS.

• **Datasets types:** It is known that the type of a dataset and the places of cities can affect the method. In another expression, the number of cities and the way they are placed and distributed have a big influence on the performance of the algorithm.

The word “known datasets” means that these examples are taken from the TSPLIB [32], which is a website that contains a library of many TSP datasets and is approved by most of the researchers in this problem. On the other hand, there are other examples that are designed and developed by the researchers themselves to be used separately in order to be compatible with their methods. The range of cities that used by the researchers in each method are illustrated in table (1).

• **Summary of comparison:** In general, finding the optimal solution of a TSP is complicated [29]. Therefore, many researchers have been trying to develop and design competitive algorithms as well as modify and hyper well-known algorithms such as GA, ACO, and PSO.

Regarding the accuracy of solutions, 34 methods have been studied and classified as the following: "ABC, ACO, CSO, DBA, GA, EDFA, MHuS, PPA, PSO, and TS" were able to reach an approximated solution and they are available online in different languages and programs as shown in table (1). On the other hand, "CHA, CRO, DA, DIWO, DMBO, GWO, SGO, SMO, WFA, Galaxy based search algorithm, and Man-Machine Approach" that their codes are not provided online have reached an approximated solution as well as most of them have overcome the well-known methods that are mentioned above.

Researchers of "EVOA and CHA" have not touched on many comparisons with the well-known algorithms, but they have presented their new methods only. Moreover, researchers of "BFOA and IIOA" have compare only the time spent during solving the TSP with the well-known methods above.

Researchers of DSSO, DCOA, DCSO, HBMO, LISA, TSPBMA, and the scientific algorithm claimed that they reached an optimal solution in their experimental. However, no codes of these methods are available online.

Finally, it is worth to mention that the two methods "B&B and dynamic programing" are considered two of the most important methods to find an optimal solution. A brief summary is shown in table (1) below.

Table 1: The classification of algorithms in terms of codes availability online. As well as programs and languages in which the codes are written.

No.	Algorithm Name	Ref.	Year	Dataset	Available programming language
1	Dynamic programming	6]	962	1-21	C#,C++,Java, MATLAB®, Python
2	Branch and Bound (B&B)	[24]	1963	10-40	C#,C++,Java, MATLAB®, Python
3	Man-Machine Approach	[22]	1971	≤ 200	-
4	Genetic Algorithm (GA)	[15]	1985	10-666	C++, Cuda, Java, JavaScript, MATLAB®, Python
5	Tabu Search (TS)	[14]	1989	42-75	C++, Fortran, Java, MATLAB®, Python
6	Ant Colony Optimization (ACO)	[11]	1992	30	C++, Java, JavaScript, MATLAB®, Python
7	Particle Swarm Optimization (PSO)	[41]	2003	14	C++, MATLAB®, Python
8	Immune Inspired-based Optimization Algorithm (IIOA)	[4]	2007	10-1000	-
9	TSP Biogeography Migration Algorithm (TSPBMA)	[28]	2010	30-130	-
10	Cockroach Swarm Optimization (CSO)	[10]	2011	30	Java
11	Artificial Bee Colony (ABC)	[21]	2011	150 - 200	C++, Python
12	Honey Bees Mating Optimization (HBMO)	[25]	2011	51-85900	-

13	Evolutionary Discrete Firefly Algorithm (EDFA)	[20]	2011	16-666	C, C#, JavaScript, Julia
14	Discrete Cat Swarm Optimization (DCSO)	[8]	2013	51-280	-
15	Egyptian Vulture Optimization Algorithm (EVOA)	[39]	2013	16-280	-
16	Bacterial Foraging Optimization Algorithm (BFOA)	[40]	2014	10-150	-
17	Discrete Bat-inspired Algorithm (DBA)	[34]	2014	51-198	C
18	Coral Reefs Optimization algorithm (CRO)	[35]	2014	15-400	-
19	The scientific algorithm	[13]	2014	14-300	-
20	Water Flow-like Algorithm (WFA)	[38]	2014	51-3975	-
21	Lightning Inspired Search Algorithm (LISA)	[12]	2015	51-198	-
22	Discrete Invasive Weed Optimization Algorithm(DIWO)	[45]	2015	48-2392	-
23	Discrete Cuttlefish Optimization Algorithm (DCOA)	[33]	2015	14-152	-
24	Memetic Hunting Search algorithm (MHuS)	[1]	2016	70-124	C#, Java
25	Discrete Monarch Butterfly Optimization (DMBO)	[42]	2016	50-200	-
26	Soccer Game Optimization (SGO)	[31]	2017	30-1655	-
27	Plant Propagation Algorithm (PPA)	[36]	2017	14-101	MATLAB®
28	Discrete Swallow Swarm Optimization (DSSO)	[7]	2017	51-225	-
29	Dragonfly Algorithm (DA)	[17]	2018	22-101	-
30	galaxy based search algorithm	[30]	2018	150-200	-

31	Grey Wolf Optimization (GWO)	[37]	2018	14-100	-
32	Black Hole algorithm (BH)	[18]	2018	22-101	-
33	Spider Monkey Optimization (SMO)	[3]	2019	14-101	-
34	Camels Herd Algorithm (CHA).	[2]	2019	48-14461	-

References

1. Agharghor, Amine, Mohammed Essaid Riffi, and Fayçal Chebihi. "A memetic hunting search algorithm for the traveling salesman problem." *2016 4th IEEE International Colloquium on Information Science and Technology (CiSt)*. IEEE, 2016.
2. Ahmed, Zied O., Ahmed T. Sadiq, and Hasanen S. Abdullah. "Solving the Traveling Salesman's Problem Using Camels Herd Algorithm." *2019 2nd Scientific Conference of Computer Sciences (SCCS)*. IEEE, 2019.
3. Ayon, Safial Islam, et al. "Spider Monkey Optimization to Solve Traveling Salesman Problem." *2019 International Conference on Electrical, Computer and Communication Engineering (ECCE)*. IEEE, 2019.
4. Bakhouya, Mohamed, and Jaafar Gaber. "An immune inspired-based optimization algorithm: Application to the traveling salesman problem." *Advanced Modeling and Optimization* 9.1 (2007): 105-116.
5. Bellmore, Mandell, and George L. Nemhauser. "The traveling salesman problem: a survey." *Operations Research* 16.3 (1968): 538-558.
6. Bellman, Richard. "Dynamic programming treatment of the travelling salesman problem." *Journal of the ACM (JACM)* 9.1 (1962): 61-63.
7. Bouzidi, Safaa, and Mohammed Essaid Riffi. "Discrete swallow swarm optimization algorithm for travelling salesman problem." *Proceedings of the 2017 International Conference on Smart Digital Environment*. 2017.
8. Bouzidi, Abdelhamid, and Mohammed Essaid Riffi. "Discrete cat swarm optimization to resolve the traveling salesman problem." *International Journal* 3.9 (2013).
9. Carpaneto, Giorgio, and Paolo Toth. "Some new branching and bounding criteria for the asymmetric travelling salesman problem." *Management Science* 26.7 (1980): 736-743.
10. Cheng, Le, et al. "Cockroach swarm optimization algorithm for TSP." *Advanced Engineering Forum*. Vol. 1. Trans Tech Publications Ltd, 2011.
11. Colorni, Alberto, Marco Dorigo, and Vittorio Maniezzo. "An Investigation of some Properties of an" Ant Algorithm"." *Ppsn*. Vol. 92. No. 1992. 1992.

12. El Majdouli, Mohamed Amine, and Abdelhakim Ameur El Imrani. "Lightning Inspired Search Algorithm: Introduction & application to the traveling salesman problem." *2015 10th International Conference on Intelligent Systems: Theories and Applications (SITA)*. IEEE, 2015.
13. Felipe, Denis, Elizabeth Ferreira Gouvêa Goldbarg, and Marco César Goldbarg. "Scientific algorithms for the car renter salesman problem." *2014 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2014.
14. Glover, Fred. "Tabu search—part I." *ORSA Journal on computing* 1.3 (1989): 190-206.
15. Grefenstette, John, et al. "Genetic algorithms for the traveling salesman problem." *Proceedings of the first International Conference on Genetic Algorithms and their Applications*. Vol. 160. No. 168. Lawrence Erlbaum, 1985.
16. Grötschel, Martin, and Manfred W. Padberg. "On the symmetric travelling salesman problem I: inequalities." *Mathematical Programming* 16.1 (1979): 265-280.
17. Hammouri, Abdelaziz I., et al. "A dragonfly algorithm for solving traveling salesman problem." *2018 8th IEEE International Conference on Control System, Computing and Engineering (ICCSCE)*. IEEE, 2018.
18. Hatamlou, Abdolreza. "Solving travelling salesman problem using black hole algorithm." *Soft Computing* 22.24 (2018): 8167-8175.
19. Hoffman, Karla L., Manfred Padberg, and Giovanni Rinaldi. "Traveling salesman problem." *Encyclopedia of operations research and management science* 1 (2013): 1573-1578.
20. Jati, Gilang Kusuma. "Evolutionary discrete firefly algorithm for travelling salesman problem." *International conference on adaptive and intelligent systems*. Springer, Berlin, Heidelberg, 2011.
21. Karaboga, Dervis, and Beyza Gorkemli. "A combinatorial artificial bee colony algorithm for traveling salesman problem." *2011 International Symposium on Innovations in Intelligent Systems and Applications*. IEEE, 2011.
22. Krolak, Patrick, Wayne Felts, and George Marble. "A man-machine approach toward solving the traveling salesman problem." *Communications of the ACM* 14.5 (1971): 327-334.
23. Langevin, André, François Soumis, and Jacques Desrosiers. "Classification of travelling salesman problem formulations." *Operations Research Letters* 9.2 (1990): 127-132.
24. Little, John DC, et al. "An algorithm for the traveling salesman problem." *Operations research* 11.6 (1963): 972-989.
25. Marinakis, Yannis, Magdalene Marinaki, and Georgios Dounias. "Honey bees mating optimization algorithm for the Euclidean traveling salesman problem." *Information Sciences* 181.20 (2011): 4684-4698.
26. Mataija, Mirta, Mirjana Rakamarić Šegić, and Franciska Jozić. "Solving the travelling salesman problem using the Branch and bound method." *Zbornik Veleučilišta u Rijeci* 4.1 (2016): 259-270.
27. Mavrovouniotis, Michalis, Mien Van, and Shengxiang Yang. "Pheromone modification strategy for the dynamic travelling salesman problem with weight changes." *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE, 2017.
28. Mo, Hongwei, and Lifang Xu. "Biogeography migration algorithm for traveling salesman problem." *International Conference in Swarm Intelligence*. Springer, Berlin, Heidelberg, 2010.

29. Papadimitriou, Christos H., and Kenneth Steiglitz. "Some examples of difficult traveling salesman problems." *Operations Research* 26.3 (1978): 434-443.
30. Phu-ang, Ajchara. "The new technique based on the galaxy based search algorithm for solving the symmetric travelling salesman problem." *2018 International ECTI Northern Section Conference on Electrical, Electronics, Computer and Telecommunications Engineering (ECTI-NCON)*. IEEE, 2018.
31. Purnomo, Hindriyanto Dwi, et al. "Soccer game optimization for travelling salesman problem." *2017 International Conference on Innovative and Creative Information Technology (ICITech)*. IEEE, 2017.
32. Reinelt, Gerhard. "TSPLIB—A traveling salesman problem library." ORSA journal on computing 3.4 (1991): 376-384.
33. Riffi, Mohammed Essaid, and Morad Bouzidi. "Discrete cuttlefish optimization algorithm to solve the travelling salesman problem." *2015 Third World Conference on Complex Systems (WCCS)*. IEEE, 2015.
34. Saji, Yassine, Mohammed Essaid Riffi, and Belaïd Ahiod. "Discrete bat-inspired algorithm for travelling salesman problem." *2014 Second World Conference on Complex Systems (WCCS)*. IEEE, 2014.
35. Salcedo-Sanz, S., et al. "The coral reefs optimization algorithm: a novel metaheuristic for efficiently solving optimization problems." The Scientific World Journal 2014 (2014).
36. Selamoğlu, Birsen İ., and Abdellah Salhi. "The plant propagation algorithm for discrete optimisation: The case of the travelling salesman problem." *Nature-inspired computation in engineering*. Springer, Cham, 2016. 43-61.
37. Sopto, Dibbendu Singha, et al. "Modified Grey Wolf Optimization to Solve Traveling Salesman Problem." *2018 International Conference on Innovation in Engineering and Technology (ICIET)*. IEEE, 2018.
38. Srour, Ayman, Zulaiha Ali Othman, and Abdul Razak Hamdan. "A water flow-like algorithm for the travelling salesman problem." *Advances in Computer Engineering* 2014 (2014).
39. Sur, Chiranjib, Sanjeev Sharma, and Anupam Shukla. "Solving travelling salesman problem using Egyptian vulture optimization algorithm–a new approach." *Intelligent Information Systems Symposium*. Springer, Berlin, Heidelberg, 2013.
40. Verma, Om Prakash, Rashmi Jain, and Vindhya Chhabra. "Solution of travelling salesman problem using bacterial foraging optimisation algorithm." *International Journal of Swarm Intelligence* 1.2 (2014): 179-192.
41. Wang, Kang-Ping, et al. "Particle swarm optimization for traveling salesman problem." *Proceedings of the 2003 international conference on machine learning and cybernetics (IEEE cat. no. 03ex693)*. Vol. 3. IEEE, 2003.
42. Wang, Gai-Ge, et al. "A discrete monarch butterfly optimization for Chinese TSP problem." *International Conference on Swarm Intelligence*. Springer, Cham, 2016.
43. Yahia, Warif B., Mohammed W. Al-Neama, and Ghassan E. Arif. "A HYBRID OPTIMIZATION ALGORITHM OF ANT COLONY SEARCH AND NEIGHBOUR-JOINING METHOD TO SOLVE THE TRAVELLING SALESMAN PROBLEM."

44. Yaseen, Mustafa T., Ali Hasan Ali, and Ibrahim A. Shanan. "Weighted (k, n)-arcs of Type (nq, n) and Maximum Size of (h, m)-arcs in PG(2, q). Communications in Mathematics and Applications 10.3 (2019): 361-368.
45. Zhou, Yongquan, et al. "A discrete invasive weed optimization algorithm for solving traveling salesman problem." *Neurocomputing* 151 (2015): 1227-1236.