The Maximum Scatter Travelling Salesman Problem: A Hybrid Genetic Algorithm

Zakir Hussain Ahmed [1, *], Asaad Shakir Hameed [2], Modhi Lafta Mutar [2], Mohammed F. Alrifaie [3], Mundher Mohammed Taresh [4]

[1]Department of Mathematics and Statistics, College of Science, Imam Mohammad Ibn Saud Islamic University (IMSIU), Riyadh, Kingdom of Saudi Arabia.

[2]Department of Mathematics, General Directorate of Thi-Qar Education, Ministry of education, Thi-Qar, Iraq.

[3]Department of Information and Communications, Basra University College of science and technology, Basrah, Iraq.

[4]College of Information Science and Engineering, Hunan University, Chang Sha, China.

[*]*Corresponding: zaahmed@imamu.edu.sa*

Abstract
In this paper, we consider the maximum scatter traveling salesman problem (MSTSP), a travelling salesman problem (TSP) variant. The problem aims to maximize the minimum length edge in a salesman's tour that travels each city only once in a network. It is a very complicated *NP*-hard problem, and hence, exact solutions can be found for small sized problems only. For large-sized problems, heuristic algorithms must be applied, and genetic algorithms (GAs) are found to be very successfully to deal with such problems. So, this paper develops a hybrid GA (HGA) for solving the problem. Our proposed HGA uses sequential sampling algorithm along with 2-opt search for initial population generation, sequential constructive crossover, adaptive mutation, randomly selected one of three local search approaches, and the partially mapped crossover along with swap mutation for perturbation procedure to find better quality solution to the MSTSP. Finally, the suggested HGA is compared with a state-of-art algorithm by solving some TSPLIB symmetric instances of many sizes. Our computational experience reveals that the suggested HGA is better. Further, we provide solutions to some asymmetric TSPLIB instances of many sizes.

Keywords:
Hybrid genetic algorithm; maximum scatter traveling salesman problem; sequential constructive crossover; adaptive mutation; local search; perturbation procedure.

1. Introduction

The travelling salesman problem (TSP) is a popular problem, which finds smallest tour of a salesman that starts journey from a headquarters city and visits all outstanding n cities (nodes) only once and then comes back to the headquarters. The TSP is a NP- Hard problem [1] and several good procedures are suggested to solve the problem. However, some circumstances need different constraints to accept a tour as solution. One such constraint is to maximize the least cost edge in a tour of the salesman, which is called the maximum scatter TSP (MSTSP). So, the MSTSP finds a Hamiltonian cycle/circuit so as to maximize the least cost edge. That means, each city in the Hamiltonian circuit is far from (scattered) its preceding and succeeding cities. The problem is also known as the max-min 1-neighbour TSP. In general, the max-min m-neighbour TSP aims to maximize the least cost between a city and its all m-neighbor cities in the Hamiltonian cycle/circuit. The bottleneck TSP (BTSP) is very close to the MSTSP. The BTSP finds a Hamiltonian circuit so as to minimize the maximum cost edge [2]. Further, the maximum TSP (MaxTSP) which finds a Hamiltonian cycle/circuit so as to maximize the length of any tour is also closely related to the MSTSP [3].

Let us formally define the MSTSP as follows: Let a network with n cities (city 1 is the headquarters) and an nXn distance (time or cost, etc.) matrix D=[d_{ij}] associated with ordered pair (i, j) of cities is given. Let (1=α_0, α_1, α_2,....,α_{n-1} , α_n=1) ≡ {1→α_1→α_2→.... →α_{n-1}→1} be a tour. The tour value is defined as $\min\{d_{\alpha_i,\alpha_{i+1}}: i = 0, 1, 2, \ldots, n-1\}$. The problem is to maximize the tour value.

The problem may be converted to the BTSP by supposing c_{ij} = L-d_{ij}, where C = [c_{ij}]$_{nxn}$ is equivalent BTSP's distance (or cost) matrix and L is a very big number [4]. The MSTSP was first defined in [5], which has several applications ([1], [6]). The problem is NP-hard [1], and no polynomial-time algorithm is available for solving the problem. So, finding optimal solution for large-sized problem instances using exact method is not possible. Thus, for finding better solution, within acceptable computational effort, to such type of problems, generally, heuristic/metaheuristic algorithms are applied. Tabu search [7], simulated annealing [8], ant colony algorithm [9], insertion heuristic [10], variable neighbourhood method [11], genetic algorithms [12], etc., are some popular metaheuristic algorithms. Among them, genetic algorithms

Manuscript received June 5, 2023
Manuscript revised June 20, 2023
https://doi.org/10.22937/IJCSNS.2023.23.6.24

(GAs) are widely applied algorithms, and so, we are applying GAs to solve the MSTSP.

Genetic Algorithms (GAs) are based on mimicking the Darwinian *survival-of-the-fittest* theory in the natural biology [12]. They are very robust metaheuristics that can solve large-sized problems quickly. They were effectively applied to various complex optimization problems for solving them. For any problem, each feasible solution may be encoded as a string called chromosome whose value is its objective function [13]. Simple GAs start from a chromosome set called initial population and then go through mainly three basic operations – selection, crossover, and mutation to generate better populations in following generations. Selection operator probabilistically copies some chromosomes to the following generation. Crossover arbitrarily selects two parent chromosomes and mates them to produce offspring chromosome(s). Mutation picks out a position at a chromosome randomly and changes its value. The crossover along with selection is the major process in GA search. Mutation varies the search space and defends genetic material losses. Thus, crossover probability is set to be very high, whereas mutation probability is set to be very low [14]. As crossover operator is very important operator, so, using better crossover operators can achieve better GAs. Normally, crossover methods that were applied for the TSP are proposed to apply to its variations also. A computational experience carried amongst eight crossover operators for the MSTSP proven that sequential constructive crossover (SCX) is the best operator [15].

Though simple GAs using three basic operators can solve complex optimization problems quickly, but very often they converge prematurely, and get trapped in local minima [13]. So, one must apply some techniques to overcome premature convergence issue and to enhance the solution obtained by simple GAs. So, this paper develops a hybrid GA (HGA) for finding solution to the MSTSP. Our proposed HGA uses sequential sampling algorithm along with 2-opt search for initial population generation, sequential constructive crossover, swap mutation, randomly selected one of three local search approaches, and the partially mapped crossover along with an adaptive mutation for perturbation procedure to find better quality solution to the MSTSP. Generally, perturbation procedure is used to overcome premature convergence issue. Finally, our HGA is compared against multi-start iterated local search (MS-ILS(h_1+h_2)) [16] by solving some TSPLIB symmetric instances of different sizes. Our experimental investigation demonstrates that the HGA is one of the best algorithms. Further, we report solutions to some asymmetric TSPLIB instances of several sizes.

This paper is arranged as follows: A literature survey for the MSTSP is provided in Section 2. Section 3 develops a hybrid genetic algorithm for the problem, while Section 4 reports computational experience of the proposed algorithm. Finally, Section 5 provides conclusion and forthcoming research works.

2. Literature Review

The MSTSP is a difficult NP-hard problem. Methods to solve this kind of optimization problems are grouped into two broad groups – exact and heuristic methods ([17]-[18]). There are very less literatures on the MSTSP. The first procedure for solving the problem is developed by Arkin et al. [1]. They proved that the problem is NP-hard, and no constant-factor approximation procedure can be devised unless P = NP. A factor-2 (claimed to be best) approximation procedure is developed for the max-min 1-neighbour TSP with the triangle inequality for both path and cycle adaptations. Further, they developed procedures for the max-min 2-neighbour TSP with the triangle inequality for both the cycle and path adaptations. Finally, the procedures extended to find an approximation solution of the max-min m-neighbour TSP for path version.

Approximation procedures for the max-min 2-neighbour TSP with the triangle inequality was developed by Chiang [19] for the cycle and path adaptations by improving the procedures in [1]. As reported, both procedures are very simple. Some studies on the MSTSP and its related versions are reported by John [6].

An approximation procedure for the MSTSP with the triangle inequality was developed by Kabadi and Punnen [20] that claimed to find the best bound for this case.

An improved procedure of the procedure in [1] was proposed for the points on a line to a regular mXn-grid by Hoffmann et al. [14] that claimed to obtain optimal solutions. They further claimed that the procedure takes linear computational effort to obtain optimal tour in some cases.

The multi-salesmen MSTSP called multiple MSTSP (MMSTSP) was proposed by Dong et al. [21]. They proposed three improved GAs for the problem. Their improved algorithms used greedy initialization, simulated annealing, and hill-climbing algorithms. As reported, their algorithms are effective algorithms that can expose various characteristics to find solution of the problem.

In [16], a multi-start iterated local search procedure was developed for the MSTSP. Based on modified 2-opt moves and insertion, two local search procedure were proposed in their procedure. As reported, their algorithm found very good results on some symmetric TSPLIB instances.

In [15], eight GAs were developed using eight crossover methods for the MSTSP. A comparative study was reported on some asymmetric and symmetric TSPLIB instances. It was showed that the sequential constructive crossover (SCX) is the best, greedy crossover (GX) is the worst and partially mapped crossover (PMX) is the second-best.

It is mentioned that the BTSP is very close to the MSTSP. Lexisearch approaches were developed for the BTSP in ([22], [23]). Further hybrid algorithms were developed for the BTSP in ([25],[26]). The MaxTSP is also close to the MSTSP for which a hybrid GA is developed for finding solution to the problem [24].

3. A Hybrid Genetic Algorithm for the MSTSP

Genetic algorithms (GAs) are established to be effective for the traditional TSP and its some variants. Though they do not assure the optimality of their obtained solutions, they normally obtain very close optimal solutions rapidly. In this section, we develop a hybrid GA (HGA) for solving the MSTSP.

3.1. Initial Population

The first job in GAs is to determine a chromosome representation procedure for representing solutions of a problem so that GA operators can produce feasible chromosome(s). For TSP and its variants, mainly path representation is used which lists cities so that no city is duplicated in a chromosome. We consider this path representation for the MSTSP. As an example, let {1, 2, 3, 4, 5, 6, 7, 8} be the cities in an 8-city problem, and the chromosome (1, 3, 2, 7, 8, 6, 4, 5) represents the tour {1→3→2→7→8→6→4→5 →1} whose objective as well as fitness function is the least cost among the edges in this tour. As starting with a better initial population can give better solution quality quickly. We use sequential sampling approach [22] [24] for generating initial population for our HGA, that was successfully applied on other TSP variants ([23]-[24]) ([25]-[26]). Since this approach cannot search all space, so, to improve the initial population, we apply 2-opt search to every chromosome for enhancing the population. However, if the newly obtained chromosome is better than the old one, replace it by the new one, otherwise, no action is taken. Due to the strong capability of 2-opt local search, it can improve the search space of our proposed algorithm.

3.2. Selection Operator

In selection operation, no new chromosome is created, only some of the fitter chromosomes are passed to the breeding pool for the subsequent operation/generation. By selecting a greater section of fitter chromosomes, this operation simulates the Darwinian hypothesis of survival-of-the-fittest in biology. Normally, the proportionate selection is used where a chromosome is chosen depending on its probability of selection. We use stochastic remainder selection procedure [27] for the proposed HGA.

3.3. Crossover Operator

Crossover operator performs a very big role in GAs, where two parent chromosomes as well as a crossover point within the chromosomes' length are selected and the information of the chromosomes after the crossover point are exchanged. Quite a few good crossover methods are present in the literature for the traditional TSP that can be applied for the MSTSP. Ahmed [15] applied eight crossover operators, namely, ordered crossover [28], partially mapped crossover [29], cycle crossover [30], alternating edges crossover [31], generalized N crossover [32], greedy crossover [31], edge recombination crossover [33], sequential constructive crossover [13] on the MSTSP, and reported a comparative study among them. As reported, sequential constructive crossover (SCX) is observed as the best method. We also apply this SCX in our proposed HGA. The steps of SCX algorithm are as follows [15]:

Step 1: Start from 'city 1' (i.e., current city p =1).

Step 2: Search sequentially both parent chromosomes and take the first 'legitimate city' (the city which is not yet visited) emerged after 'city p' in both parents. If no 'legitimate city' after 'city p' is present in any parent chromosome, search from the first city in chromosome and take the first 'legitimate city' and go to Step 3.

Step 3: Suppose 'city α' and 'city β' are in 1st and 2nd parents correspondingly, then for choosing the following city go to Step 4.

Step 4: If $c_{p\alpha} > c_{p\beta}$, then choose 'city α', otherwise, 'city β' as the subsequent city and merge it to the incomplete offspring. If this offspring is a full chromosome, then stop, else, the present city is renamed as 'city p' and go to Step 2.

Sometimes SCX creates bad offspring. So, to maintain a mixture of offspring and parent in a population, we replace the 1st parent by the offspring if it is better. In addition, the 2-opt local search is used on the better offspring to improve it further. Since the SCX operator produces only an offspring. So, to keep population size same in all generations, when selecting next pair for crossover, the present 2nd parent will be selected as the 1st parent and the 3rd chromosome will be as the 2nd parent, and so on.

3.4. Mutation Operator

As some weaker chromosomes are omitted in selection and crossover processes in any generation, so, there might be some stronger chromosomes' structures which were lost forever. So, normally, mutation is applied to regain them. In traditional mutation operations, a gene (or a position) is chosen arbitrarily in a chromosome and then alters its subsequent allele (city). Some of the mutation operators are inversion mutation, insertion mutation, swap mutation, adaptive mutation [34]. The adaptive mutation is implemented for our HGA. To perform this mutation, the data from all chromosomes in a population are collected to detect a pattern amongst them. If the mutation is to be performed, then the chromosomes that do not match the pattern will be muted. The steps of adaptive mutation are as follows:

Step 1: Consider all chromosomes in the current population.

Step 2: Create a one-dimensional array of size n (the problem size), suppose, A, by storing a city (gene) that appears least number of times in the current position of all chromosomes.

Step 3: If mutation is allowed, two genes are selected randomly so that they are not same in the corresponding positions of the array, A, and they are exchanged.

3.5. Local Search

There are various local search procedures available in the literature, amongst them combined mutation is seen as a nice local search procedure ([2], [25], [26]). It merges insertion, inversion, and swap mutations with 1.00 probabilities. Insertion mutation selects a city (gene) in a chromosome and then inserts into an arbitrary position. Inversion mutation selects two points in a chromosome and inverts the sub-chromosome between them. Swap mutation selects two cities (genes) arbitrarily and exchanges them. We define these three mutations as local search procedures in our HGA as follows. Suppose (α_1, α_2, α_3, ..., α_n) be a chromosome, then the insertion mutation may be defined as:

Step 0: For i: = 1 to n-1 do the following steps.

Step 1: For j: = i+1 to n do the following steps.

Step 2: If inserting location α_i after location α_j reduces the cost of the assignment, then insert the location α_i after the location α_j.

The inversion mutation may be defined as:

Step 0: For i: = 1 to n-1 do the following steps.

Step 1: For j: = i+1 to n do the following steps.

Step 2: If inverting substring between the locations α_i and α_j reduces the present assignment cost, then invert the substring.

The swap mutation may be defined as:

Step 0: For i: = 1 to n-1 do the following steps.

Step 1: For j: = i+1 to n do the following steps.

Step 2: If swapping the locations α_i and α_j reduces the present assignment cost, then swap them.

In our local search procedure, one of these three mutations is selected arbitrarily in our HGA for the MSTSP.

3.6. Perturbation Procedure

Though GAs are very good methods, but sometimes, they get stuck in local optima. This may be due to identical population, and so, the population have to be varied. Perturbation procedure is useful in escaping from local optima. If (Best Solution – Average Solution) < 0.20*Best Solution, then we apply partially mapped crossover (PMX), swap mutation and combined mutation operators. The PMX selects two crossover points, describes swap mappings in the segment between these points, and delivers two offspring. Further, mutation can assist other operators to beat local optima issue and thus, can find better solutions.

3.7. Hybrid GA

Hence, for the MSTSP, our proposed HGA is presented in Figure 1.

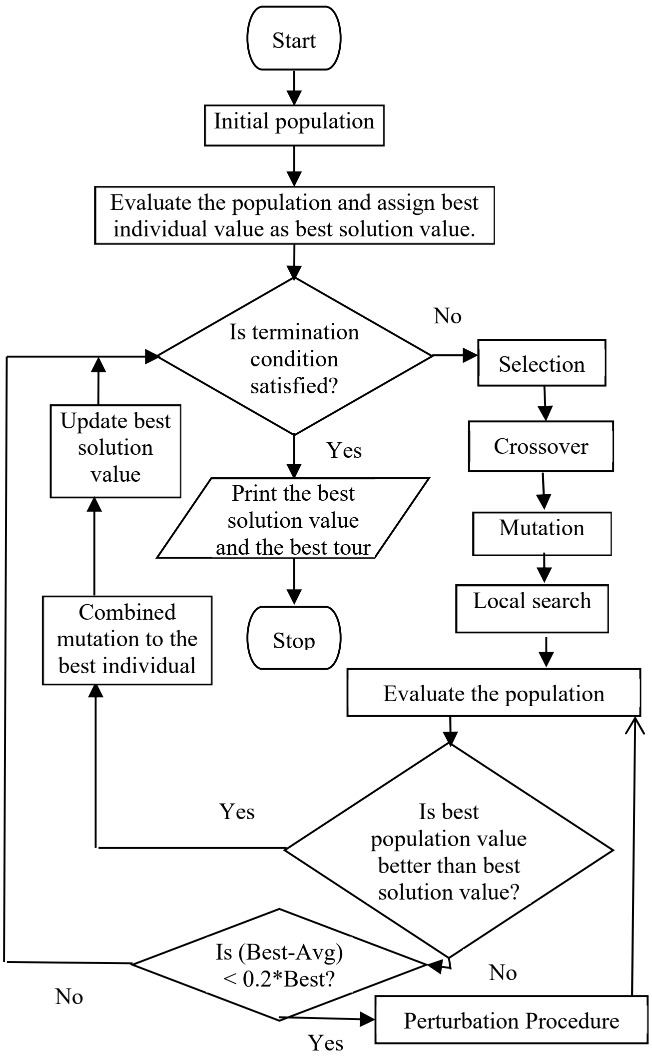

Fig. 1. Flow-chart of our HGA.

We encoded our proposed HGA in Visual C++. To determine the value of HGA, computational experience is performed on some typical TSPLIB instances [35] of many sizes and then implemented on a Laptop with i7-1065G7 CPU@1.30 GHz and 8 GB RAM under MS Windows 10. We run HGA for separate parameter settings, and chosen parameters are recorded in Table 1.

Table 1. Parameters for the HGA

Parameters	**Values**
Population size	50
Crossover probability	100%
Mutation probability	10%
Termination criterion	2,00 generations
No. of runs for each instance	20 times

We assess our projected HGA with a state-of-art algorithm, namely, multi-start iterated local search (MS-ILS(h_1+h_2)) [16] on some TSPLIB symmetric instances of sizes from 14 to 100. We record best solution (BS), worst solution (WS), average solution (AS), and computational time (Time) (in seconds) for each problem instance in Table 2. Better solutions are shown in boldfaces.

Table 2. Comparative study of MS-ILS(h_1+h_2) and HGA for symmetric TSPLIB instances

Instance	n	MS-ILS(h_1+h_2)				HGA			
		BS	WS	AS	Time	BS	WS	AS	Time
burma14	14	498	498	498.00	0.03	498	498	498.00	0.00
ulysses16	16	677	677	677.00	0.03	**726**	**726**	**726.00**	0.04
gr17	17	239	239	239.00	0.04	**257**	**257**	**257.00**	0.05
gr21	21	370	370	370.00	0.06	370	370	370.00	0.01
ulysses22	22	687	687	687.00	0.06	**726**	**726**	**726.00**	0.12
gr24	24	164	164	164.00	0.07	**173**	**169**	**169.20**	0.20
fri26	26	102	102	102.00	0.09	**103**	**103**	**103.00**	0.18
bayg29	29	189	189	189.00	0.11	**195**	**195**	**195.00**	0.23
bays29	29	231	221	230.00	0.11	**234**	**231**	**232.20**	0.33
dantzig42	42	73	71	72.60	0.21	**75**	**71**	**73.40**	0.59
swiss42	42	129	124	**128.30**	0.22	129	126	127.60	1.38
att48	48	1103	1103	1103.00	0.27	1103	1103	1103.00	1.49
gr48	48	558	545	**555.40**	0.29	558	554	555.10	1.25
hk48	48	1098	1089	1095.80	0.28	1098	1094	1095.80	2.08
eil51	51	39	39	**39.00**	0.31	39	35	37.44	2.13
berlin52	52	541	541	541.00	0.32	541	541	541.00	1.10
brazil58	58	1906	1906	1906.00	0.39	**1959**	**1930**	**1939.70**	3.69
st70	70	63	63	**63.00**	0.54	63	62	62.08	4.41
eil76	76	41	41	**41.00**	0.66	41	38	39.12	8.94
pr76	76	9214	9214	9214.00	0.70	9214	9214	9214.00	3.42
gr96	96	4778	4756	4763.50	1.09	**4817**	**4778**	**4795.20**	22.33
rat99	99	111	111	**111.00**	1.19	111	99	101.90	26.94
kroA100	100	2101	2101	2101.00	1.08	2101	2101	2101.00	10.48
kroB100	100	1935	1933	1934.20	1.11	1935	1933	**1934.30**	10.52
kroC100	100	2253	2230	2242.00	1.25	2253	2237	**2242.30**	11.61
kroD100	100	2067	2027	**2047.10**	1.22	2067	2024	2044.30	10.83
kroE100	100	2002	1977	1995.60	1.11	2002	1977	**1996.80**	10.27
rd100	100	672	672	672.00	1.09	672	672	672.00	11.12

Looking at best and average solutions, for the ten instances, namely, ulysses16, gr17, ulysses22, gr24, fri26, bayg29, bays29, dantzig42, brazil58 and gr96, our HGA could find better solutions than solutions found by MS-ILS(h_1+h_2). Looking at only average solutions, for other seven instances, namely, swiss42, gr48, eil51, st70, eil76, rat99 and kroD100, MS-ILS(h_1+h_2) is better, and for other ten instances, namely, kroB100, kroC100 and kroED100, our HGA is better. For remaining instances, both algorithms are equally performing. Overall, looking at best and average solutions, our HGA is better than MS-ILS(h_1+h_2). We further depict best solutions by HGA and MS-ILS(h_1+h_2) in the Figure 2, which also shows that our algorithm HGA is better than MS-ILS(h_1+h_2). In addition, for the unreported problem instances of sizes more than 100, both algorithms obtain same best solutions. However, MS-ILS(h_1+h_2) takes less computational time. So, looking at the solution quality, our suggested HGA is seen better.

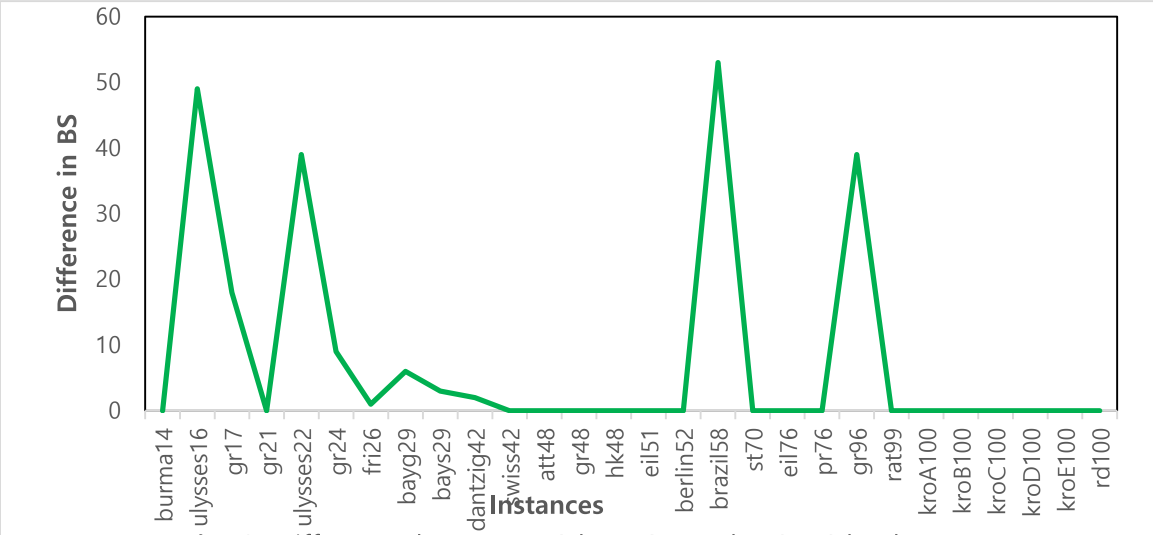

Fig. 2: Difference between BS by HGA and MS-ILS(h_1+h_2)

We further, report solutions by HGA for some asymmetric TSPLIB instances of sizes from 17 to 443 in Table 3. To our best of knowledge, no literature reported solutions of these instances. So, we could not perform comparative study.

Table 3. Summary of results by HGA for asymmetric TSPLIB instances

Instance	n	BS	WS	AS	Time
br17	17	5	5	5.00	0.00
ftv33	34	143	143	143.00	0.56
ftv35	36	154	153	153.30	0.71
ftv38	39	154	151	152.60	1.09
p43	43	17	17	17.00	0.02
ftv44	45	162	158	160.85	1.59
ftv47	48	168	163	165.15	2.56
ry48p	48	1232	1207	1219.10	3.05
ft53	53	379	374	377.05	4.11
ftv55	56	154	150	152.90	4.43
ftv64	65	157	154	155.75	7.58
ft70	70	970	965	967.05	13.19
ftv70	71	157	155	156.40	16.77
kro124p	100	2347	2333	2340.12	21.03
ftv170	171	171	167	170.20	119.28
rbg323	323	20	18	19.20	158.34
rbg358	358	18	16	17.80	186.32
rbg403	403	15	15	15.00	206.54
rbg443	443	15	15	15.00	223.78

5. Conclusion & Discussions

A hybrid genetic algorithm (HGA) is developed for finding useful solution to the maximum scatter traveling salesman problem (MSTSP). As beginning with a better initial population, GA can lead faster convergence of the solution, a sequential sampling algorithm along with 2-opt search is used for generating initial population. About the other GA operators, sequential constructive crossover, adaptive mutation, randomly chosen one of three local search approaches, and the partially mapped crossover along with swap mutation for perturbation procedure to find better quality solution to the problem.

Computational experience on some TSPLIB symmetric instances indicate the value of HGA. Out of 28 instances, for 10 instances HGA could find new solutions. Also, comparative analysis shows that HGA could touch best solutions by multi-start iterated local search (MS-ILS(h_1+h_2)) at least once in twenty runs. So, our suggested HGA is showed to be better. Though HGA is found to be the better, however, it takes more computational time than by MS-ILS(h_1+h_2). Hence, a better local search and perturbation procedure may obtain better solution quality, which is under the investigation.

Acknowledgment

This research was supported by Deanery of Academic Research, Imam Muhammad Ibn Saud Islamic University, Saudi Arabia vide Grant No. 18-11-09-010. The first author thanks the Deanery for its financial support.

References

[1] E.M. Arkin, Y.-J. Chiang, J.S.B. Mitchell, S.S. Skiena, and T.-C. Yang, On the maximum scatter traveling salesperson problem, SIAM Journal of Computing 29 (1999) 515–544.
[2] Z.H. Ahmed, A hybrid genetic algorithm for the bottleneck traveling salesman problem. ACM Transactions on Embedded Computing Systems 12 (2013) Art. No. 9.
[3] A. Barvinok, S.P. Fekete, D.S. Johnson, A. Tamir, G.J. Woeginger and R. Woodroofe, The geometric maximum traveling salesman problem, Journal of the ACM 50(5) (2003) 641–664.
[4] J. LaRusic and A.P. Punnen, the asymmetric bottleneck traveling salesman problem: Algorithms, complexity and empirical analysis, Computers & Operations Research 43 (2014) 20–35.
[5] F. Scholz, Coordination hole tolerance stacking, Technical Report BCSTECH-93-048, Boeing Computer Services, November 1993.
[6] L.R. John, the bottleneck traveling salesman problem and some variants, Master of Science of Simon Fraser University, Canada, 2010.
[7] W.B. Carlton and J.W. Barnes, Solving the travelling salesman problem with time windows using tabu search, IEE Transaction 28 (1996) 617–629.
[8] J.W. Ohlmann and B.W. Thomas, A compressed-annealing heuristic for the traveling salesman problem with time windows, INFORMS Journal of Computing 19 (1) (2007) 80–90.
[9] C.-B. Cheng and C.-P. Mao, A modified ant colony system for solving the travelling salesman problem with time windows, Mathematical Computer Modelling 46 (2007) 1225–1235.
[10] M. Gendreau, A. Hertz, G. Laporte and M. Stan, A generalized insertion heuristic for the traveling salesman problem with time windows, Operations Research 46 (3) (1998) 330–335.
[11] R.F. da Silva and S. Urrutia, A general VNS heuristic for the traveling salesman problem with time windows, Discrete Optimization 7 (4) (2010) 203–211.
[12] DE. Goldberg. Genetic algorithms in search, optimization, and machine learning, Addison-Wesley, New York, 1989.
[13] Z.H. Ahmed, Genetic algorithm for the traveling salesman problem using sequential constructive crossover operator, International Journal of Biometrics & Bioinformatics 3 (2010) 96-105.
[14] I. Hoffmann, S. Kurz, and J. Rambau, The maximum scatter TSP on a regular grid, in Operations Research Proceedings 2015, Springer, 2017, pp. 63–70.
[15] Z.H. Ahmed, A comparative study of eight crossover operators for the maximum scatter travelling salesman problem, International Journal of Advanced Computer Science and Applications (IJACSA) 11 (2020) 317-329.
[16] P. Venkatesh, A. Singh and R. Mallipeddi, A multi-start iterated local search algorithm for the maximum scatter traveling salesman problem, 2019 IEEE Congress on Evolutionary Computation (CEC), Wellington, New Zealand, 2019, pp. 1390-1397.
[17] A.S. Hameed, B.M. Aboobaider, N.H. Choon, M.L Mutar, W.H. Bilal. 'Review on the Methods to Solve Combinatorial Optimization Problems Particularly: Quadratic Assignment Model', International Journal of Engineering & Technology, 7, pp. 15–20. 2018.
[18] M.L. Mutar, M.A. Burhanuddin, A.S. Hameed, N. Yusof, H.J. Mutashar. 'An efficient improvement of ant colony system algorithm for handling capacity vehicle routing problem', International Journal of Industrial Engineering Computations, 11(4), pp. 549–564. 2020. DOI: 10.5267/j.ijiec.2020.4.006.
[19] Yi-J. Chiang. New approximation results for the maximum scatter TSP. Algorithmica, 41 (2005) 309–341.
[20] S.N. Kabadi and A.P. Punnen. The bottleneck TSP, In the Traveling Salesman Problem and Its Variations, G. Gutin and A.P. Punnen (eds.), Chapter 15, Kluwer Academic, Dordrecht, 2002.
[21] W. Dong, X. Dong and Y. Wang, The improved genetic algorithms for multiple maximum scatter traveling salesperson problems, In J. Li et al. (Eds.): CWSN 2017, CCIS 812, pp. 155–164, 2018.
[22] Z.H. Ahmed, A lexisearch algorithm for the bottleneck travelling salesman problem, International Journal of Computer Science and Security 3(5) (2010) 569-577.

Manuscript received June 5, 2023
Manuscript revised June 20, 2023
https://doi.org/10.22937/IJCSNS.2023.23.6.24

[23] Z.H. Ahmed, A data-guided lexisearch algorithm for the bottleneck travelling salesman problem, International Journal of Operational Research 12(1) (2011) 20-33.
[24] Z.H. Ahmed, A hybrid sequential constructive sampling algorithm for the bottleneck traveling salesman problem, International Journal of Computational Intelligence Research 6(3) (2010) 475-484.
[25] Z.H. Ahmed, A hybrid genetic algorithm for the bottleneck traveling salesman problem, ACM Transactions on Embedded Computing Systems (TECS)12(1) (2013) 1-10.
[26] Z.H. Ahmed, An experimental study of a hybrid genetic algorithm for the maximum traveling salesman problem, Mathematical Sciences 7(1) (2013) 1-7.
[27] K. Deb, Optimization for engineering design: algorithms and examples, Prentice Hall of India Pvt. Ltd., New Delhi, India, 1995.
[28] L. Davis, Job-shop scheduling with genetic algorithms, Proceedings of an International Conference on Genetic Algorithms and Their Applications, 136-140, 1985.
[29] D.E. Goldberg and R. Lingle, Alleles, loci and the travelling salesman problem, In J.J. Grefenstette (ed.) Proceedings of the 1st International Conference on Genetic Algorithms and Their Applications. Lawrence Erlbaum Associates, Hilladale, NJ, 1985.
[30] I.M. Oliver, D. J. Smith and J.R.C. Holland, A Study of permutation crossover operators on the travelling salesman problem, In J.J. Grefenstette (ed.). Genetic Algorithms and Their Applications: Proceedings of the 2nd International Conference on Genetic Algorithms. Lawrence Erlbaum Associates, Hilladale, NJ, 1987.
[31] J. Grefenstette, R. Gopal, B. Rosmaita and D. Gucht, Genetic algorithms for the traveling salesman problem, In Proceedings of the First International Conference on Genetic Algorithms and Their Applications, (J. J. Grefenstette, Ed.), Lawrence Erlbaum Associates, Mahwah NJ, 160–168, 1985.
[32] N.J. Radcliffe and P.D. Surry, Formae and variance of fitness, In D. Whitley and M. Vose (Eds.) Foundations of Genetic Algorithms 3, Morgan Kaufmann, San Mateo, CA, 51-72, 1995.
[33] D. Whitley, T. Starkweather and D. Shaner, the traveling salesman and sequence scheduling: quality solutions using genetic edge recombination, In L. Davis (Ed.) Handbook of Genetic Algorithms. Van Nostrand Reinhold, New York, 350-372, 1991.
[34] Z.H. Ahmed, an improved genetic algorithm using adaptive mutation operator for the quadratic assignment problem, 38th International Conference on Telecommunications and Signal Processing 2015 (TSP 2015) (2015) 1-5.
[35] G. Reinelt, TSPLIB, http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/