The Minimum Latency Problem: A Hybrid Genetic Algorithm

Zakir Hussain Ahmed

Department of Computer Science, Al Imam Mohammad Ibn Saud Islamic University (IMSIU), P.O. Box No. 5701, Riyadh-11432, Kingdom of Saudi Arabia

Abstract

This paper presents a hybrid genetic algorithm to solve the minimum latency problem (MLP). The problem is a variation of the well-known travelling salesman problem (TSP) in which sum of arrival times at the nodes is minimized. The problem arises in many real-life applications such as logistics for relief supply, scheduling and data retrieval in computer networks. The computational results on TSPLIB instances show the efficiency of our proposed algorithm. Finally, a comparative study is carried out against an existing state-of-art algorithm to establish the goodness of our algorithm. The study shows the effectiveness of our proposed hybrid algorithm.

Key words:

Minimum latency problem; hybrid genetic algorithm; local search; NP-hard; sequential constructive crossover.

1. Introduction

The minimum latency problem (MLP) was introduced in 1967 [1] that can be defined as follows. Let G = (V, A) be a complete directed graph, where V = {1, 2, …, n} is the set of nodes in the network (graph) and A = {(i, j): i, j ϵ V, i≠j} is the set of arcs with corresponding travel time t(i, j). Node 1represents the depot while other nodes represent customers. The problem seeks a Hamiltonian circuit that minimizes $\sum_{i=1}^{n} l(i)$, l(i) represents the latency of a node i ϵ V, that is, the total time to visit i. Two versions of the problem are considered in the literature – one finding a Hamiltonian path starting from the depot node and another finding a Hamiltonian circuit starting from and ending at the depot node. We consider the later one that the circuit must start and end at the depot.

The problem is a variation of the well-known traveling salesman problem (TSP), is also known as delivery man problem [2], cumulative TSP [3] and school bus driver problem [4], traveling repairman problem [5] in which a repairman is supposed to visit the nodes of a network in a way to minimize the overall waiting times of the customers located in the nodes. The problem has many real-life applications - one of them is a home delivery of pizzas [2]. In this application, several delivery orders are clustered, and someone hopes to minimize the delivery time. Other applications include in the computer networks where someone hopes to find information stored somewhere in the networks [6], single-machine scheduling problem with sequence-dependent processing times that aims to minimize the total flow time of the tasks [1].

Despite the clear similarities to the usual TSP, the MLP seems to be much less researched [7]. The differences between these problems is that in the MLP the objective is to minimize the total latency time of all the customers, while in the usual TSP the objective is to minimize the total travel time of a single salesman. The MLP is more complex because it includes complex objective instead of a single travel time objective, as in the original TSP. The problem was proved to be NP-hard for general metric spaces [8]. For the weighted graph G with six nodes, represented by time matrix Tin Table 1, the optimal solution for the TSP is 81 =(9+10+17+21+12+12) with optimal tour {1→5→6→3→4→2→1}, whereas the MLP solution is 259 = (9 + (9+10) + (9+10+12) + (9+10+12+19) + (9+10+12+19+17) + (9+10+12+19+17+16)) = (9*6 + 10*5 + 12*4 + 19*3 + 17*2 + 16*1) with optimal tour {1→5→4→2→3→6→1}.

Table 1: Time matrix T

	1	2	3	4	5	6
1	9999	12	39	42	9	16
2	12	9999	19	12	32	15
3	39	19	9999	21	45	17
4	42	12	21	9999	10	16
5	9	32	45	10	9999	10
6	16	15	17	16	10	9999

Various exact and heuristic algorithms are found to solve this problem. Since, the problem is NP-hard, it is very difficult to solve optimally, and in fact, to date no optimal algorithm has been found that can solve the problem in polynomial time. So, heuristic algorithms have been used to solve the problem. Out of heuristics, genetic algorithms (GAs) are successfully implemented to such kind of combinatorial optimization problems [9].

We are going to use GA for obtaining heuristically optimal solution to the problem. Selection, crossover and mutation are three basic operators in GA, of which crossover is the most important operator. So, numerous crossover operators have been developed for solving various optimization problems. In this paper, we use the sequential constructive crossover (SCX) [10] operator for the problem. Then a GA based on SCX is developed for solving the problem, which is then compared with GA using generalized n-point crossover (GNX) for some benchmark instances reported

Manuscript received November 5, 2018
Manuscript revised November 20, 2018

in TSPLIB website. Experimental results show that SCX operator is better than GNX. Then our GA is hybridized by incorporating with a local search and an immigration algorithm to enhance the solution quality, and finally a comparative study is carried out of our hybrid algorithm against a state-of-art heuristic algorithm [11].
This paper is organized as follows: Section 2 presents a literature review, while Section 3 describes our proposed hybrid genetic algorithm for the MLP. Computational results are presented in Section 4. Finally, Section 5 presents discussion and concluding remarks.

2. Literature review

Various exact and heuristic algorithms are developed to solve the MLP in the literature. Still, exact algorithms are very limited to small sized problem instances, while few competent heuristic algorithms are developed.
Lucena [12] developed an exact enumerative algorithm that depends on a non-linear integer formulation where lower bounds are calculated using a Lagrangian relaxation. Bianco [3] developed two exact algorithms which incorporate lower bounds using a Lagrangian relaxation. Fischetti et al. [2] developed an enumerative algorithm that uses lower bounds using a linear integer programming formulation. Méndez-Díaz et al. [13] proposed Mixed Integer Programming (MIP) formulations that also introduced various valid inequalities which are evaluated using a branch-and-cut algorithm. Bigras et al. [14] proposed some integer programming formulations along with a branch-and-bound algorithm. Ezzine et al. [15] developed two integer programming formulations for the problem. Abeledo et al. [16-17] proposed branch-cut-and-price algorithm, which solves the problem instances of size up to 107.
The first approximation algorithm was proposed by Blum et al. [18] with an approximation factor of 144. Chaudhuri et al. [4] proposed an approximation algorithm with the smallest approximation factor of 3.59 for general metric spaces. Archer and Blasiak [19] proposed an algorithm that obtained the smallest approximation factor of 3.03 for an edge-weighted tree.
Very few metaheuristic algorithms are available in the literature for the MLP. Dewilde et al. [20] developed a tabu search (TS) for the MLP with profits. Ngueveu et al. [21] developed a memetic algorithm that produces high quality solutions on the some MLP benchmark instances. Salehipour et al. [11] developed a heuristic algorithm based on greedy randomized adaptive search procedure (GRASP) along with variable neighborhood descent (VND) and variable neighborhood search (VNS) procedures and reported the computational results on some benchmark instances.

3. Our proposed hybrid genetic algorithm

In GAs, first solutions are encoded as feasible chromosomes (or individuals) such that the genetic operators result in feasible chromosomes. A simple GA begins by generating set of chromosomes (initial population), and then goes through some genetic operators to create new, and possibly, better populations as following generations.
We consider the order representation for a chromosome that lists the label of nodes. Assume a solution v = (v1, v2, …, vn) with v1 being the first and last node in the solution. The total latency for this solution can be calculated as follows, which is our objective function that aims to be minimized.

$$f(v) = \sum_{i=1}^{n} (n - i + 1) t(v_i, v_{i+1})$$

Note that nodes vn+1≡v1 and t(vi, vi+1) is the travel time between nodes vi and vi+1. With respect to the time matrix in Table 1, value of the tour a tour {1→5→4→2→3→6→1}, represented as (1, 5, 4, 2, 3, 6), is 259.The fitness function is the inverse of the objective function for our problem. We first generate randomly an initial population of defined population size, apply stochastic remainder selection [22] and then apply following methods for our hybrid GA.

3.1 Sequential constructive crossover operator

Crossover is the most important operator in GAs in which one (or two) offspring chromosome(s) is (are) produced by applying to pair of selected parent chromosomes. There are several crossover operators for the TSP which also can be applied to the MLP. However, we are going to use the sequential constructive crossover [10] (SCX) which has been successfully applied to usual TSP and its some variations [23-26]. Efficiency of the SCX will be compared with generalized n-point crossover (GNX). Our SCX operator for the MLP is as follows:

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

Step 2: Sequentially search both the parent chromosomes and consider the first 'legitimate node' appeared after 'node p' in each parent. If no 'legitimate node' after 'node p' is present in any of the parent, search sequentially from the beginning of the chromosome and consider the first 'legitimate node', and go to Step 3.

Step 3: Suppose the 'node α' and the 'node β' are found in 1st and 2nd parent respectively, then for selecting the next node go to Step 4.

Step 4: If t(p, α)<t(p, β), then select 'node α', otherwise, 'node β' as the next node and concatenate it to

the partially constructed offspring chromosome. If the offspring is a complete chromosome, then stop, otherwise, rename the present node as 'node p' and go to Step 2.

Let a pair of selected parent chromosomes be P: (1, 5, 3, 6, 4, 2) and Q: (1, 6, 2, 4, 3, 5) with values 431 and 381 respectively using Table 1. Using the SCX we get offspring chromosome, O, as (1, 5, 6, 2, 4, 3) with value 263 = (6*6 + 10*5 + 15*4 + 12*3 + 21*2 + 39*1), which is less than both parents.

3.2 Adaptive Mutation Operator

Mutation is an operator that randomly changes some genes in the chromosomes. We consider adaptive mutation [27] with mutation probability (Pm) where data from all chromosomes in the current population is collected to detect a pattern among them. If the mutation takes place, then chromosomes that do not resemble the pattern will be muted. This operator is proposed for the quadratic assignment problem and found very good results. The algorithm for our problem can be described as follows:

Step 1: Consider all chromosomes in the current population.

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

Step 3: If mutation is allowed, select randomly two genes such that they are not same in the corresponding positions of the array, A, and swap them.

Generally, the mutation probability is set very low, whereas the crossover probability is set very high. A simple GA repeatedly applies (possibly) three operators, namely, selection, crossover and mutation, till either the population converges or till reaches the maximum number of generations (iterations). A simple GA which incorporates local search is called hybrid GA (HGA). We are considering following local search method for our HGA.

3.3 Local search

We consider a local search method which is a combined mutation operation [23] that combines three mutation operators – insertion, inversion and reciprocal exchange, with cent percentage of probabilities. Suppose P[1], P[2], …, P[n] be a chromosome, then the local search algorithm is as follows:

Step 1: For i: = 1 to n-2 do the following steps.

Step 2: For j: = i+1 to n-1 do the following steps.

Step 3: If inserting node P[i] after node P[j] reduces the present tour value, then insert the node P[i] after the node P[j]. In any case go to step 4.

Step 4: If inverting substring between the nodes P[i] and P[j] reduces the present tour value, then invert the substring. In any case go to step 5.

Step 5: If swapping the nodes P[i] and P[j] reduces the present tour value, then swap them.

3.4 Immigration

To improve competency of GAs, the population must be diversified. For this purpose, an immigration method is implemented, where some new chromosomes are inserted to the population after some generations. In this work, 20% of the population is replaced in random places using sequential constructive sampling [23-25] for next generation.

Our hybrid genetic algorithm (HGA) may be summarized as in Figure 1. The next section presents computational experience for our HGA for the MLP and shows efficiency of the algorithm as against two existing heuristic algorithms on TSPLIB instances [28].

3.5 Lower bound

To measure the quality of the solution, Salehipour et al. [11] used a lower bound for the problem which is a variant of the minimum spanning tree (MST). It is computed by sorting the edges of the MST of the graph in order of increasing weight and multiplying each edge with a factor like the edges of the MLP solution. Since our problem is to find a circuit rather than a path, hence we consider the last edge twice; by that number of edges will be n, as many number of nodes in the network. Then the smallest edge is multiplied by n, the second-smallest by $n - 1$, so on and the largest edge is multiplied with a factor 1. For finding MST we use the well-known Kruskal's algorithm. Using the Table 1, MST will be having the edges bearing their weights in ascending order as 9, 10, 10, 12, 17. For our problem, 17 will be considered twice. So, the lower bound is

LB = 9*6 + 10*5 + 10*4 + 12*3 + 17*2 + 17*1 = 231.

4. Computational results

For comparing the effectiveness of the crossover operators, simple GAs using GNX (GA-GNX) and SCX (GA-SCX) have been encoded in Visual C++ and run on a Laptop with Intel(R) Core(TM) i3-3217 with CPU @ 1.80GHz and 4.00GB RAM under MS Windows 7. In the experiments, some benchmark instances from TSPLIB were used. The experiments were performed twenty times for each instance. Population size, crossover probability,

mutation probability, and termination criterion are the parameters of GAs. We set 1.0 (i.e., 100%) as crossover probability to have a clear picture of crossover operators, 0.02 (i.e., 2%) as mutation probability, 50 as population size, and 20000 generations for simple GAs and 500 for HGA as a termination criterion.

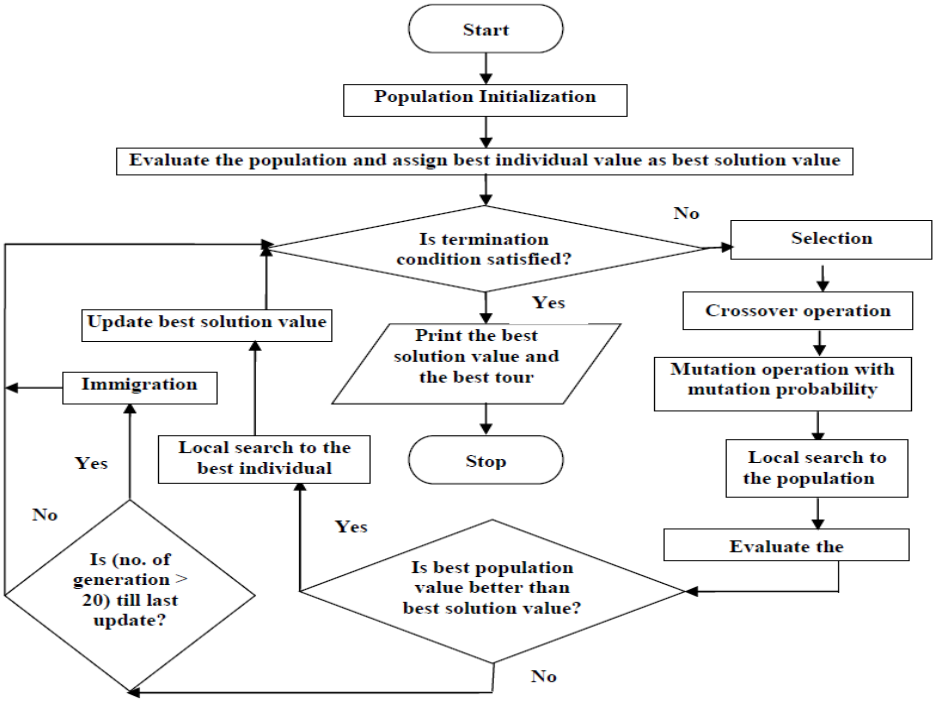

Fig. 1 Flow-chart of our hybrid genetic algorithm.

Table 2 shows comparative study of the simple GAs using two crossover operators (i.e., GA-GNX and GA-SCX) for the twelve asymmetric TSPLIB instances from ftv33 to ftv70. The solution quality is measured by the percentage of gap (%) of the obtained solution (SOL) to the lower bound (LB), which is computed by the formula: Gap (%) = 100*(SOL-LB)/SOL. So, we report gap (%) of best solution and average solution to the LB of 20 runs. In terms of solution quality, GA-SCX outperforms GA-GNX. This can be seen clearly in the Figure 2.

Table 2: Summary of the results by the crossover operators for some asymmetric benchmark TSPLIB instances

Instance	LB	GA-GNX		GA-SCX	
		Best Sol Gap (%)	Avg Sol Gap (%)	Best Sol Gap (%)	Avg Sol Gap (%)
ftv33	11922	49.62	53.32	39.87	41.63
ftv35	13679	49.45	51.03	40.03	42.43
ftv38	15891	47.14	51.67	39.42	40.27
p43	2402	79.92	80.58	79.03	79.03
ftv44	19336	45.76	53.42	41.46	42.51
ftv47	20813	53.51	56.20	44.83	45.81
ry48p	228801	35.15	37.74	25.85	27.29
ft53	63112	65.70	69.41	60.41	62.30
ftv55	23282	56.46	58.67	41.79	42.77
ftv64	29825	59.48	61.99	43.06	44.96
ft70	1007952	27.16	27.91	22.08	22.97
ftv70	35036	59.63	61.86	44.12	45.11
Average		52.42	55.32	43.50	44.76

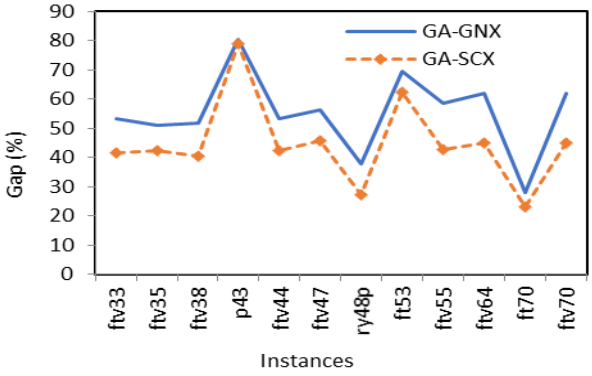

Fig. 2 Percentage of solution gap by the GA using two crossover operators for some benchmark instances

The solution quality of our proposed GA (i.e., GA-SCX) is enhanced by hybridizing with a local search and an immigration method. Table 3 shows the result of our HGA on the above problem instances. Since, to our knowledge, no any literature provides solution for the above instances, hence we cannot judge the quality of the obtained solutions. However, in terms of the percentage of average gap to the lower bound, it is seen that HGA could find very good solutions, if not optimal, to the tested asymmetric problem instances.

Table 3. Summary of the results by our HGA for some asymmetric instances

Instance	LB	BestSol	Gap (%)	AvgSol	Gap (%)
ftv33	11922	19387	38.51	19387.00	38.51
ftv35	13679	22811	40.03	22811.00	40.03
ftv38	15891	25498	37.68	25498.00	37.68
p43	2402	11452	79.03	11452.00	79.03
ftv44	19336	32039	39.65	32691.30	41.68
ftv47	20813	35984	42.16	36013.76	42.24
ry48p	228801	299144	23.51	299323.60	23.57
ft53	63112	152818	58.70	156599.20	61.18
ftv55	23282	37673	38.20	38735.95	41.02
ftv64	29825	50001	40.35	51536.85	43.42
ft70	100795	21277970	21.13	1294149.60	22.39
ftv70	35036	60741	42.32	62131.60	44.61
ftv170	151597	220749	31.33	221386.12	31.61
Average			40.97		42.08

Table 4. Comparative study between GRASP+ [11] and our HGA

Instance	GRASP+	HGA	
	BestSol	BestSol	AverageSol
st70	19553	19215	19323.15
rat99	56994	54984	56234.20
kroD100	976830	949594	969899.50
lin105	585823	585823	591853.15
pr107	1983475	1980767	1982084.55
rat195	213371	210191	214022.15
pr226	7226554	7100308	7198027.00
lin318	5876537	5560679	5798076.85
pr439	18567170	17688561	18700275.15

HGA is then compared with an existing state-of-art heuristics found in the literature, namely, GRASP+VNS/VND [11] (GRASP+) for some symmetric TSPLIB instances. Table 4 reports best solution obtained using GRASP+ and the best solution and average solution on 20 runs by our HGA. In terms of solution quality, it is found that best solutions obtained by our HGA are better than GRASP+ for eight instances. For the instance lin105, both algorithms obtained same solution. Overall our HGA is found to be better than GRASP+.

5. Conclusion and discussion

In this study, a hybrid genetic algorithm is developed to solve minimum latency problem. For that we used the sequential constructive crossover (SCX) to propose a simple genetic algorithm (GA) for solving the problem (MLP). We presented a comparative study between SCX and generalized n-point crossover (GNX) for some benchmark TSPLIB instances. In terms of quality of the solution, SCX is found to be far better than the GNX. Then a hybrid GA (HGA) is developed by incorporating a local search and an immigration method to our simple GA. Then our proposed HGA was used to compare its performance with those of GRASP +VNS/VND [11] algorithm as an existing state-of-art heuristic found in the literature. The computational results reveal that our HGA is found to be better than GRASP +VNS/VND for the tested instances.

Acknowledgements

The author is thankful to the anonymous honourable reviewers for their valuable comments and suggestions.

References

[1] Conway R, Maxwell W, Miller, L. Theory of Scheduling. Addison-Wesley, 1967.

[2] Fischetti M, Laporte G, Martelo S. The delivery man problem and cumulative methods. Operations Research, 1993, 41: 1055–1064.

[3] Bianco L, Mingozzi A, Ricciardelli S. The traveling salesman problem with cumulative costs, Networks, 1993, 23: 81-91.

[4] Chaudhuri K, Godfrey B, Rao S, Talwar K. Paths, trees and minimum latency tours. IEEE Symposium on Foundations of Computer Science-FOCS, 2003: 36-45.

[5] Tsitsiklis JN. Special cases of traveling salesman and repairman problems with time windows. Networks, 1992, 22: 263–282.

[6] Ausiello G, Leonardi S, Marchetti-Spaccamela A. On salesman, repairmen, spiders and other traveling agents. In Proceeding of the Italian Conference on Algorithms and complexity, 1994: pp. 1-16.

[7] Goemans M, Kleinberg J. An improved approximation ratio for the minimum latency problem, Mathematical Programming, 1998, 82: 111-124.

[8] Sahni S, Gonzalez T. P-complete approximation problems. Journal of the ACM, 1976, 3: 555-565.

[9] Goldberg DE. Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley, New York, 1989.

[10] Ahmed ZH. Genetic algorithm for the traveling salesman problem using sequential constructive crossover operator. International Journal of Biometrics & Bioinformatics, 2010, 3: 96-105.

[11] Salehipour A, Sorensen K, Goos P, Braysy O. Efficient GRASP+VND and GRASP+VNS metaheuristics for the traveling repairman problem. 4OR: A Quarterly Journal of Operations Research, 2011, 9: 189–209.

[12] Lucena A. Time-dependent traveling salesman problem - The deliveryman case. Networks, 1990, 20: 753–763.

[13] Méndez-DíazI, Zabala P, Lucena A. A new formulation for the traveling deliveryman problem. Discrete Applied Mathematics, 2008, 156: 3223–3237.

[14] Bigras L-P, Gamache M, Savard G. The time-dependent traveling salesman problem and single machine scheduling problems with sequence dependent setup times. Discrete Optimization, 2008, 5: 685–699.

[15] Ezzine IO, Semet F, Chabchoub H. New formulations for the traveling repairman problem. In: Proceedings of the 8th International Conference of Modeling and Simulation, 2010.

[16] Abeledo H, Fukasawa R, Pessoa A, Uchoa E. The time dependent traveling salesman problem: Polyhedra and algorithm. Technical Report, RPEP 10, Universidade Federal Fluminense, Brasil, 2010.

[17] Abeledo H, Fukasawa R, Pessoa A, Uchoa E. The time dependent traveling salesman problem: Polyhedra and branch-cut-and-price algorithm. In: Proceedings of the 9th

International Symposium on Experimental Algorithms, SEA, 2010, pp. 202–213.
[18] Blum A, Chalasanit P, Pulleyblankt B, Raghavan P, Sudan M. The minimum latency problem. In: Proceedings of the 26th Annual ACM Symposium on Theory of Computing, 1994, pp. 163–171.
[19] Archer A, Blasiak A. Improved approximation algorithms for the minimum latency problem via prize-collecting strolls. In: Proceedings of the 21st Annual ACM-SIAM Symposium on Discrete Algorithms, 2010, pp. 429–447.
[20] Dewilde T, Cattrysse D, Coene S, Spieksma FCR, Vansteenwegen P. Heuristics for the traveling repairman problem with profits. In: Proceedings of the 10th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems, ATMOS, 2010, pp. 34–44.
[21] Ngueveu S, Prins C, Calvo RW. An effective memetic algorithm for the cumulative capacitated vehicle routing problem. Computers & Operations Research, 2010, 37: 1877–1885.
[22] Deb K. Optimization for Engineering Design: Algorithms and Examples. Prentice Hall of India Pvt Ltd, New Delhi, India, 1995.
[23] Ahmed ZH. A hybrid genetic algorithm for the bottleneck traveling salesman problem. ACM Transactions on Embedded Computing Systems, 2013, 12: Art. No. 9.
[24] Ahmed ZH. An experimental study of a hybrid genetic algorithm for the maximum travelling salesman problem. Mathematical Sciences, 2013, 7: 1-7.
[25] Ahmed ZH. The ordered clustered travelling salesman problem: A hybrid genetic algorithm. The Scientific World Journal, Art ID 258207, 2014, 13 pages.
[26] Ahmed ZH. A simple genetic algorithm using sequential constructive crossover for the quadratic assignment problem. Journal of Scientific & Industrial Research, 2014, 73: 763-766.
[27] Ahmed ZH. An improved genetic algorithm using adaptive mutation operator for the quadratic assignment problem. Proceedings of 38th International Conference on Telecommunications and Signal Processing 2015 (TSP 2015), 1–5, IEEE.
[28] TSPLIB. Available at: http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/