Journal of Advances in Computer Research
Quarterly ISSN: 2008-6148
Sari Branch, Islamic Azad University, Sari, I.R.Iran
(Vol. 3, No. 3, August 2012), Pages: 75-84
www.jacr.iausari.ac.ir

Solving the Traveling Salesman Problem by an Efficient Hybrid Metaheuristic Algorithm

Mohammad Ahmadvand[1]✉, Majid Yousefikhoshbakht[2], Narges Mahmoodi Darani[1]

(1) Department of mathematics, Malayer Branch, Islamic Azad University, Malayer, Iran
(2) Young Researchers & Elites Club, Hamedan Branch, Islamic Azad University, Hamedan, Iran

ahmadvand@iau-malayer.ac.ir; yosefikhoshbakht@gmail.com; n.mahmoodi@malayer.ac.ir

Received: ; Accepted:

Abstract

The traveling salesman problem (TSP) is the problem of finding the shortest tour through all the nodes that a salesman has to visit. The TSP is probably the most famous and extensively studied problem in the field of combinatorial optimization. Because this problem is an NP-hard problem, practical large-scale instances cannot be solved by exact algorithms within acceptable computational times. So, an efficient hybrid metaheuristic algorithm called ICATS is proposed in this paper. The first stage of the ICATS is to solve the TSP by the imperialist competitive algorithm (ICA), and then the TS is used for improving solutions. This process avoids the premature convergence and makes better solutions. Computational results on several standard instances of TSP show efficiency of the proposed algorithm compared with the genetic algorithm (GA), bee colony optimization (BCO), and particle swarm optimization (PSO).

Keywords: *Tabu search, Imperialist competitive algorithm, Traveling salesman problem, NP-hard problems*

1. Introduction

Combinatorial problems in mathematics are among the most difficult and time-consuming to solve. They often involve arranging sets of numbers in fairly irregular patterns in order to find extremes represented in the numbers, e.g. maxima and minima. The traveling salesman problem (TSP) is a well-known problem in the area of network and operation research. The simplicity of this problem and its complexity has attracted the attention of many researchers over a long period of time. Their importance relies upon the fact that they are difficult to be solved but are intuitively used for modeling several real world problems [1].

A general definition of the TSP is the following. Consider a set V of nodes, and a set A of arcs fully connecting the nodes V. Let c_{ij} be the length of the arc $(i,j) \in A$, that is the distance between nodes i and j. The TSP is the problem of finding a minimal length Hamiltonian circuit on the graph G= (V, A), where a Hamiltonian circuit of a graph G is a closed tour visiting once and only once all the n=|V | nodes of G, and its length is given by the sum of the lengths of all the arcs of which it is composed [2].

In practice, the basic TSP is extended with constraints, for instance, on the allowed capacity of the salesman, length of the route, arrival, departure and service time, time of

collection and delivery of goods. The extended classes of Distance Constrained TSP (DCTSP), TSP with Time Windows (TSPTW), Backhauls TSP (TSPB), Pickup and Delivery TSP (TSPPD), and Simultaneous Pickup and Delivery TSP (TSPSPD). The main goal in all TSP problems is to obtain the minimal transportation cost.

Solving the TSP was an interesting problem during recent decades. Almost every new approach for solving engineering and optimization problems has been tested on the TSP as a general test bench. First steps in solving the TSP were exact methods. Exact approaches such as Lagrangean relaxation [3] and branch and bound [4] are successfully used only for relatively small problem sizes but they can guarantee optimality based on different techniques. These techniques use algorithms that generate both a lower and an upper bound on the true minimum value of the problem instance. If the upper and lower bound coincide, a proof of optimality is achieved.

Although the TSP is conceptually simple, it is difficult to obtain an optimal solution. In an n-city situation, any permutation of n nodes yields a possible solution. As a consequence, n! possible tours must be evaluated in the search space. In other words, when the problem size is increased, the exact methods cannot solve it. So, heuristic such as gravitational emulation search [5], partitioning approach [6] and Lin-Kernighan [7] are used for solving them and settle for the suboptimal solutions in a reasonable amount of time for instances with large size. Besides, some algorithms based on greedy principles such as nearest neighbor, and spanning tree can be introduced as efficient solving methods.

Classical methods including exact and heuristic algorithms for solving the TSP usually result in exponential computational complexities. Hence, new methods are required to overcome this shortcoming. These methods include different kinds of optimization techniques, nature based optimization algorithms, population based optimization algorithms and etc. These methods are nowadays commonly called metaheuristics. These algorithms have more performance than exact and heuristic one, so nowadays they have been received much attention by researchers and scientists for solving combinatorial optimization problems. Because of using the randomization concept in search for finding better solutions, this group has more effectiveness for escaping from local optimum and can get more quality solutions. That is why the recent publications are all based on meta-heuristic approaches such as genetic algorithm (GA) [8], memetic algorithm [9], ant system (AS) [10] and particle Swarm optimization [11].

Recently, many researchers have found that the employment of hybridization in optimization problems can improve the quality of problem solving in compare to heuristic and meta-heuristic approaches. These algorithms such as Genetic algorithm with a local search, Genetic algorithm with sweep algorithm and nearest addition method, ant colony optimization and greedy heuristic, simulated annealing and tabu search (TS), neural networks and genetic algorithm, genetic algorithm and ant colony optimization and others have more ability for finding an optimal solution in combinatorial optimization problems. So, a hybrid metaheuristic algorithm in the name of ICATS is proposed in this paper. In this proposed algorithm, at first the Imperialist Competitive Algorithm (ICA) is used as construction algorithm for producing an acceptable solution. Then, the TS is applied to improving solution at the second stage.

In the following parts of this paper, a mathematical model of TSP is presented in Section 2. In Section 3, the basic ICA and TS and the proposed idea are especially explained. In Section 4, the proposed algorithm is compared with some of the other

algorithms on standard problems belonged in TSP library. Finally in Section 5, the conclusions are presented.

2. Traveling Salesman Problem

The TSP is one of the applied problems in the real world that consists of a number of nodes. These nodes must be visited from an arbitrary node called the depot by only one salesman. The route starting and finishing at the depot is required for salesman so each node is visited only once by the salesman (Figure 1).

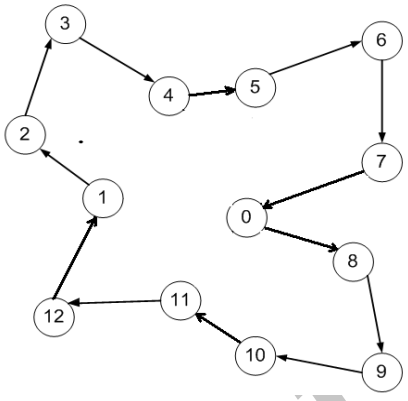

Figure1. Sample of solving TSP

The TSP can be explained mathematically as follows. Let G (V, A) be a perfect undirected connected graph with a vertex set $V = \{0,1,...,n\}$ and an edge set $A = \{(i, j) : i, j \in V, i \neq j\}$. If the graph is not perfect, the lack of any edge is replaced by an edge that has an infinite size. Besides, the solution to the TSP determines an order for nodes that obtains the minimum total cost for a salesman. In practice, minimizing the total cost is equivalent to minimizing the total distance traveled by a salesman. By introducing variables x_{ij} to represent the tour of the salesman from node i to node j, one of the integer programming formulations for the TSP can be written as:

$$\min \sum_{i=0}^{n}\sum_{j=0}^{n} c_{ij}x_{ij} \quad (1)$$

subject to

$$\sum_{i=0}^{n} x_{ij} = 1 \qquad (j = 1,...,n) \quad (2)$$

$$\sum_{j=0}^{n} x_{ij} = 1 \qquad (i = 1,...,n) \quad (3)$$

$$\sum_{i\in S}\sum_{j\in N-S} x_{ij} \geq 1 \quad (\phi \neq S \subset \{0,...,n\}, |S| \geq 2 \quad (4)$$

$$\sum_{i\in N-S}\sum_{j\in S} x_{ij} \geq 1 \quad (\phi \neq S \subset \{0,...,n\}, |S| \geq 2 \quad (5)$$

$$x_{ij} \in \{0,1\} \qquad (i = 0,...,n; j = 0,...,n) \quad (6)$$

The objective function (1) is simply to minimize the total distance traveled on a tour. Constraint set (2) ensures that the salesman arrives once at each node. Constraint set (3)

ensures that the salesman leaves each node once. Constraint sets (4) and (5) are to avoid the presence of sub-tours for a salesman. Finally, Constraint set (6) defines binary conditions on the variables. It should be noted that if the salesman travels directly from node i to node j, $x_{ij} = 1$ else $x_{ij} = 0$.

Generally, the TSP formulated is known as the Euclidean TSP, in which the distance matrix $C = [c_{ij}]_{(n+1)\times(n+1)}$ is expected to be symmetric, i.e., $c_{ij} = c_{ji}$ for all (i, j=0, 1, 2, ..., n), and to satisfy the triangle inequality, that is $c_{ik} \le c_{ij} + c_{jk}$ for all distinct ($i, j, k \in 1, 2, ..., n$).

3. Solution method

3.1 Imperialist competitive algorithm

During recent three decades, Meta-heuristic algorithms have been one of the most important groups for solving combinatorial optimization problems. These algorithms such as memetic algorithm, simulated annealing, particle swarm optimization and ant colony optimization have been successfully applied to many difficult optimization problems including TSP, vehicle routing problem, quadratic assignment problem, job-shop scheduling problem, etc.

The ICA and TS are two of the most powerful meta-heuristic algorithms that have been receiving much attention by researchers and scientists recently. The ICA introduced by Atashpaz Gargari uses socio-political evolution of human as a source of inspiration for developing a powerful optimization strategy [12]. This algorithm considers the imperialism as a level of human social evolution and by mathematically modeling this complicated political and historical process, harnesses it as a tool for evolutionary optimization. The ICA has been applied successfully in different domains namely designing controllers [13], recommender systems, characterization of elasto-plastic properties of materials [14] and many other optimization problems [15]. The results have shown great performance in both convergence rate and better global optimum achievement [16-17]. In ICA, the first step is to generate an initial population like other evolutionary algorithms. The population set includes a number of feasible solutions called a ‘country’, which corresponds to the term ‘chromosome’ in the GA. These countries are of two types: colonies and imperialists that all together form some empires. Bigger and stronger empires have more colonies than smaller and weaker ones. After forming initial empires, their colonies start moving toward their relevant imperialist country. This movement is a simple model of assimilation policy pursued by some of the imperialist states.

If one of the colonies possesses more power than its relevant imperialist, they will exchange their positions. To begin the competition between empires, total power of each empire should be calculated. It depends on the power of both an imperialist and its colonies. Imperialistic competition among these empires forms the basis of the proposed evolutionary algorithm. During this competition, weak empires collapse and powerful ones take possession of their colonies. The empire that has lost all its colonies will collapse. At last the most powerful empire will take the possession of other empires and wins the competition. In other words, imperialistic competition hopefully converges to a state in which there exists only one empire and its colonies are in the same position and have the same cost as the imperialist.

3.2 Tabu Search

The TS was firstly proposed by Glover is also very efficient for solving NP-hard problems [18]. As a local search technique, TS moves from a current solution to the best solution in its neighborhood by exploring the solution space in each iteration. The main principle of the TS method is accepting neighboring solutions that deteriorate the current objective function value in order to escape from premature local optimum. TS taboos the recently visited solutions (Tabu List) or recently applied moves for a specific number of iterations (Tabu Tenure). This tabu list has two main purposes: to prevent the return to the most recently visited solutions in order to avoid cycling; to drive the search towards the regions of the solution space not yet explored and with high potential of containing good solutions. When the search attempts to move towards a tabu declared state, this transition is forbidden, unless it improves the best solution ever encountered during the search (aspiration criterion).

A successful application of TS needs a powerful technique for search intensification and diversification. The intensification is a comprehensive exploration of some area of the solution space, typically in the neighborhood of a good solution. The diversification is leading the search to the promising regions of the solution space that has not explored yet [19].

3.3 Proposed algorithm

At the first stage of the ICATS, *n* feasible solutions and their values of the objective function are randomly generated by a bisection array shown in Figure 2. It is noted that using a random construction at this level leads to obtain solutions that have an irregular construction in feasible space. Then, *m* countries that have better objective function are selected and called imperialist countries. Furthermore, if n/m is integer number then the number of colonies randomly devoted to each empire country is equal. Otherwise the reminder countries are devoted to the imperialist with less objective function. Therefore, each imperialist with its colonies forms an empire.

1	2	3	7	5	8	4	6	20

Figure2. A Country in ICATS

After the empires are formed, each colony by absorption function increases its quality using the imperialist countries. As an example, suppose that [5 2 8 7 4 3 1 6] and [2 7 6 5 3 1 4 8] are imperialist country and its colony country respectively in the empire. Then, a random number between 2 and N-1 of the colony is selected and is arranged according to the order of the imperialist countries. For instance, if the selected number is 3, then 3 nodes from [2 7 6 5 3 1 4 8] like 7, 5 and 4 are chosen from the colony. Then, their order will be found in imperialist country which is 5, 7, and 4. Therefore, the new result for the colony will be [2 5 6 7 3 1 4 8]. It is noted that the absorption function is performed for all colonies in comparison to their imperialist countries. Each new solution and its value are replaced if the quality is increased.

At the next stage, ρ percent of colonies experience revolution. This causes variations in colonies of the empire and if possible their quality increases at each stage. The proposed method for this stage is the 3-opt local search. This algorithm is based on omitting three arcs of the tour that are not adjacent and connecting them again by another method. It can be noted that there are several routes for connecting nodes and producing the tour again, but a state that satisfies the problem's constraints is

acceptable. So, this unique tour will be accepted only if, first; the mentioned constraints are not violated, and second; the new tour produces a better value for the problem than the previous solution. Besides, omitting three arcs and again connecting them are repeated until no improving 3-opt is found. At this time, if it is a colony with better objective function than its imperialist then their situations are changed.

After this stage, the power of empires (h_j) is calculated using formula (7). In this formula, s_j is the average objective function of the colonies for empire j and λ is the parameter in [0 1] which determines the relative power of an empire compared with its imperialist. A weaker empire loses its power by losing its weakest colony to the strongest empire.

$$h_j = f_j + \lambda(s_j) \qquad j = 1,...,m \tag{7}$$

Finally, if the best solution (s^*) has been iterated five times, the algorithm ends. Otherwise, the algorithm is iterated by returning to absorption function step. After the ICA was finished, the TS starts. In the TS suppose that s^* is primitive solution and current solution s is equal with s^*. TS moves from s to the best solution in its neighborhood s' which is not in the tabu list unless it improves the best solution ever encountered during the search (aspiration criterion). The proposed TS algorithm uses two types of neighborhood moves including the insert and swap moves for s. In the insert move, a candidate node is removed from the route and inserted in the best place. On the swap move, two nodes are randomly selected and exchanged with each other. There is no direct indication in the TS that the process has converged to the best solution. There are different methods that can be used to tell the system when to stop evolving. In the proposed TS as same as ICA, if the best solution has been iterated five times, the proposed algorithm will be stopped. The Pseudo-code of the ICATS is presented in the Figure 3.

Step 1: Initialize 100 countries including 10 imperialists and 90 colonies.
Step 2: Move the colonies toward their relevant imperialist.
Step 3: Revolve 10 percent of colonies.
Step 4: If there is a colony in an empire which has better cost than the imperialist, exchange the positions of that colony and its imperialist.
Step 5: Compute the total cost of all empires.
Step 6: Pick the weakest colony from the weakest empire and give it to the best empire.
Step 7: If there is an empire with no colonies then Eliminate that.
Step 8: Save the best so far solution as s^*.
Step 9: If s^* has not been iterated five iterations, go to 2.
Step 11: Replace the current solution s by s^*.
Step 12: Construct insert and swap algorithms as N(s), tabu list T(s) and aspiration condition A(s).
Step 13: set length of T(s) is 7 and Replace the s by the best $s' \in \{N(s) - T(s) + A(s)\}$.
Step 14: If f(s') < f(s^*) then $s^* \leftarrow s'$.
Step 15: Update tabu list, neighborhood function, and aspiration condition.
Step 16: Set $s := s'$. if s^* has not been iterated five iterations, go to 12.
Step 17: End

Figure3. Pseudo-code of the ICATS

4. Computational experiments

In this section, the experimental results of the proposed algorithm on two sets of benchmark problems are shown These algorithms are applied and tested on several Euclidean sample instances of TSPLIB with sizes ranging from 24 to 229 nodes. The proposed approach was implemented in Matlab 7 language and implemented on a Pentium 4 PC at 4GHZ (2GB RAM). All TSP problem instances are obtained from TSPLIB for the symmetric TSP. The parameters of the ICATS are selected after thorough testing. A number of different alternative values were tested and the ones selected are those that gave the best computational results concerning the quality of the solution. Because the proposed approach is a meta-heuristic algorithm, the results are reported for ten independent runs, and in each run the algorithm was executed until the best solution was iterated five times.

Table 1 shows the results of the proposed algorithm for the TSP benchmark problem instances. In this table, Columns 2-6 show the problem size n, the best value result of the PA (OVR), the worst value result of the PA (OWR), the average value of the PA (AV) over the ten runs for each problem, and the best known solutions (BKS). This table shows that the PA can be used to solve the TSP effectively.

Table1. Results of ICATS for the TSP

	Instance	n	OVR	OWR	AV	BKS
1	Eil51	51	426	456	437	426
2	Pr76	76	108159	108345	108275	108159
3	Rat99	99	1211	1308	1262	1211
4	KroA100	100	21282	21451	21398	21282
5	KroB100	100	22141	22468	22326	22141
6	Rd100	100	7910	7998	7966	7910
7	Eil101	101	629	712	679	629
8	Lin105	105	14389	14599	14496	14379
9	Pr107	107	44303	44467	44398	44303
10	Pr124	124	59030	59567	59313	59030
11	Bier127	127	118282	118657	118436	118282
12	Ch130	130	6110	6245	6186	6110
13	Pr136	136	96802	97546	97342	96772
14	Pr144	144	58537	58987	58812	58537
15	Ch150	150	6535	6654	6599	6528
16	KroA150	150	26524	26989	26734	26524
17	Pr152	152	73682	73923	73889	73682
18	Rat195	195	2329	2405	2387	2323
19	D198	198	15785	16758	16454	15780
20	KroA200	200	29383	29983	29794	29368
21	Ts225	225	126940	127435	127201	126643
22	Pr226	226	80745	81279	80998	80369
23	Pr264	264	49842	50945	50123	49135
24	A280	280	2579	2795	2697	2579
25	Pr299	299	48349	48980	48539	48191

The ICATS finds the best known solutions for 15 out of 25 problems published in the literature. However, in other instances, the proposed algorithm finds nearly the best known solution, i.e. the gap is below 1.5%, and for overall the average difference is 0.12%. Besides, Figure 4 shows a comparison between the gap values of the meta-heuristic algorithms. The gap is equal to $100[c(s^{**}) - c(s^{*})]/c(s^{*}),$ where s^{**} is the best

solution found by the ICATS for a given instance, and s^* is the overall best known solution for the same instance on the Web. A zero gap indicates that the best known solution is found by the algorithm.

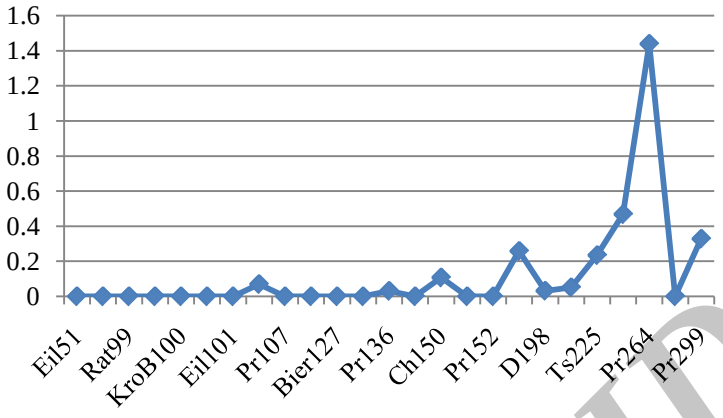

Figure5. Gap of the ICATS

Table 3 shows the results obtained for the second problem instances and presents the best results of the ICATS in comparison with some meta-heuristic algorithms including Bee Colony Optimization (BCO) [20], Particle Swarm Optimization (PSO) [21], and Genetic Algorithm (GA) [22]. The results of this comparison show that the ICATS gains equal solutions with the GA in GR24, Bayg29 and GR48, which are not large scale problems, and it gains better solutions than the GA in following scale problems such as St70 and KroA100. Furthermore, although the PSO gives an equal solution with the proposed algorithm for Berlin52, this algorithm cannot maintain this condition in the preceding examples and the ICATS yields better solutions than this algorithm for other instances including Eil51, Eil76, KroA100 and KroA200. Besides, the Computational experiments also show that in general the proposed algorithm gives better results compared to a BCO algorithm in terms of the solution's quality for each instance.

Table2. Comparison between PA and other metaheuristic algorithms

Instance	Size	GA [22]	PSO[21]	BCO[20]	ICATS	Optimum
GR24	24	1272	-	-	1272	1272
Bayg29	29	1610	-	-	1610	1610
GR48	48	5046	-	-	5046	5046
ATT48	48	-	-	10661	10628	10628
Eil51	51	-	427	428	426	426
Berlin52	52	-	7542	-	7542	7542
ST70	70	685	-	-	679	675
Eil76	76	-	540	539	538	538
KroA100	100	21504	21296	21763	21282	21282
KroB100	100	-	-	22637	22141	22141
KroC100	100	-	-	20853	20793	20749
KroD100	100	-	-	21643	21312	21294
KroE100	100	-	-	22450	22099	22068
Eil101	101	-	-	635	629	629
Lin105	105	-	-	15288	14389	14379
KroA150	150	-	-	27858	26524	26524
KroB150	150	-	-	26535	26202	26130
KroA200	200	-	29563	29961	29383	29368
KroB200	200	-	-	30350	29587	29437

In order to demonstrate the efficiency of the algorithm, some of the solutions found in the examples in table 2 are presented in Figure 5. It should be noted that in four examples presented this figure, the PA has been able to find the best solution ever found.

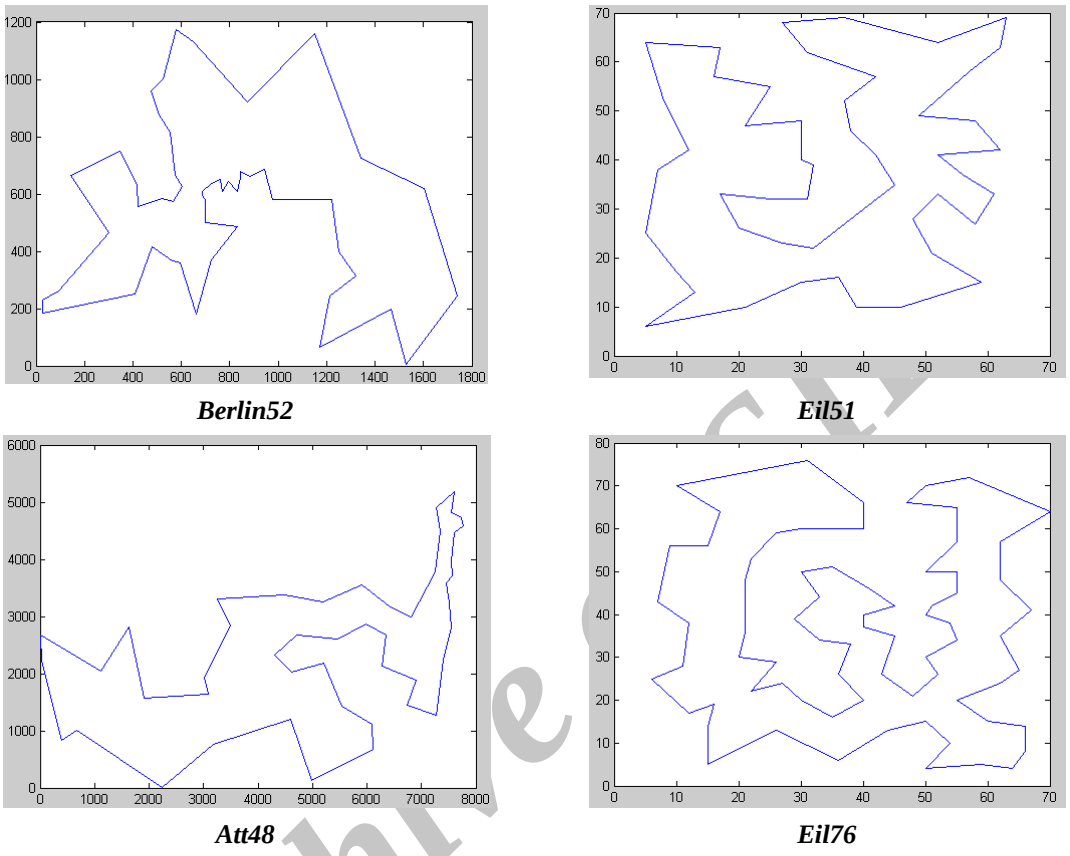

Figure6. Some of the Solutions to the TSP Found by the ICATS

5. Conclusion

In this paper, a new algorithm based on ICA and TS for solving the TSP was discussed. The ICATS is more efficient than other meta-heuristic algorithms including BCO, GA and PSO especially for large problems. Extension of this approach not only to other type of routing problems including the multiple traveling salesman problem, vehicle routing problem and school bus routing problem but also to more complex cases like assignment and scheduling problems are promising subjects for further research. The results of this research will appear in subsequent papers.

Acknowledgement

The authors would like to acknowledge the Malayer Branch, Islamic Azad University for the financial support of this work.

6. References

[1] A. S. M. Thiago, de Leandro, N. C. Neuro-immune approach to solve routing problems. Neurocomputing. 72(10-12), 2189-2197, 2009.

[2] M. Dorigo, G. Di Caro, L. M. Gambardella. Ant Algorithms for Discrete Optimization. Artificial Life, 5(3), 137-172, 1999.

[3] S. Yadlapalli, W. A. Alik, S. Darbha, M. Pachter, A Lagrangian-based algorithm for a Multiple Depot, Multiple Traveling Salesmen Problem. Nonlinear Analysis: Real World Applications. 10(4), 1990-1999, 2009.

[4] J. F. Cordeau, M. Dell'Amico, M. Iori, Branch-and-cut for the pickup and delivery traveling salesman problem with FIFO loading. Computers & Operations Research. 37(5), 970-980, 2010.

[5] S. R. Balachandar, K. Kannan, Randomized gravitational emulation search algorithm for symmetric traveling salesman problem. Applied Mathematics and Computation. 192(2), 413-421, 2007.

[6] R. M. Karp, Probabilistic analysis of partitioning algorithms for the traveling salesman problem in the plane. Mathematics of Operations Research, 2, 209-224, 1977.

[7] D. Karapetyan, G. Gutin, Lin-Kernighan Heuristic Adaptations for the Generalized Traveling Salesman Problem. European Journal of Operational Research. 208(3), 221-232, 2011.

[8] D. Kaur, M. M. Murugappan, Performance Enhancement in Solving Traveling Salesman Problem using Hybrid Genetic Algorithm. Fuzzy Information Processing Society. NAFIPS, 1–6, 2008.

[9] B. Bontoux, C. Artigues, D. Feillet, A Memetic Algorithm with a large neighborhood crossover operator for the Generalized Traveling Salesman Problem. Computers & Operations Research. 37(11), 1844-1852, 2010.

[10] S. Ghafurian, N. Javadian, An ant colony algorithm for solving fixed destination multi-depot multiple traveling salesmen problems. Applied Soft Computing. 11(1), 1256-1262, 2011.

[11] W. Zhong, J. Zhang, W. Chen, A novel discrete particle swarm optimization to solve traveling salesman problem. Evolutionary Computation. 3283–3287, 2007.

[12] E. Atashpaz-Gargari, C. Lucas, Imperialist competitive algorithm: an algorithm for optimization inspired by imperialistic competition, In: Proceedings of the IEEE Congress on Evolutionary Computation, Singapore, 2007.

[13] R. Rajabioun, F. Hashemzadeh, E. atashpaz-Gargari, B. Mesgari, F. R. Salmasi, Decentralized PID Controller Design for a MIMO Evaporator Based on Colonial Competitive Algorithm. in 17th IFAC World congress, Seoul, Korea, 2008.

[14] B. Oskouyi, E. Atashpaz-Gargari, N. Soltani, C. Lucas, Application of Imperialist Competitive Algorithm for materials property characterization from sharp indentation test. International Journal of Engineering Simulation. 2008.

[15] A. Khhabbazi, E. atashpaz, E. Lucas, Imperialist competitive algorithm for minimum bit error rate beamforming. international journal of Bio-Inspired computation. 1, 125-133, 2009.

[16] H. Frigui, R. Krishnapuram, A Robust Competitive Clustering Algorithm with Application in Computer Vision. IEEE T Patt. Anal. Machine Intell. 21(1), 450-465, 1999.

[17] A. k. Jain, P. Duin, M. jianchang, statical pattern recognition: a review, IEEE Trans. Patt. Anal. Machine Intell. 22(1), 4-37, 2000.

[18] F. Glover, Future Paths for Integer Programming and Links to Artificial Intelligence, Computers and Operations Research , 13, 533-549, 1986.

[19] C. D. Tarantilis, C. T. Kiranoudis, A flexible adaptive memory-based algorithm for real-life transportation operations: two case studies from dairy and construction sector. European Journal of Operational Research. 179, 806–822, 2007.

[20] L. P. Wong, M. Y. H. Low, C. S. Chong, A bee colony optimization algorithm for traveling salesman problem. AICMS. 818–823, 2008.

[21] W. Zhong, J. Zhang, W. Chen, A novel discrete particle swarm optimization to solve traveling salesman problem. Evolutionary Computation. 3283–3287, 2007.

[22] S. S. Ray, S. Bandyopadhyay, S.K. Pal, New operators of genetic algorithms for traveling salesman problem. Proceedings of the 17th International Conference on Pattern Recognition. 2, 497–500, 2004.