Applied Soft Computing 30 (2015) 484–490

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

A new hybrid method based on Particle Swarm Optimization, Ant Colony Optimization and 3-Opt algorithms for Traveling Salesman Problem

Mostafa Mahi [a], Ömer Kaan Baykan [b], Halife Kodaz [b,*]

[a] Department of Computer Engineering and Information Technology, Payame Noor University, Tehran, Iran
[b] Department of Computer Engineering, Faculty of Engineering, University of Selcuk, Konya, Turkey

ARTICLE INFO

Article history:
Received 11 July 2014
Received in revised form
12 November 2014
Accepted 30 January 2015
Available online 11 February 2015

Keywords:
Particle Swarm Optimization
Ant Colony Optimization
3-Opt algorithm
Traveling Salesman Problem

ABSTRACT

The Traveling Salesman Problem (TSP) is one of the standard test problems used in performance analysis of discrete optimization algorithms. The Ant Colony Optimization (ACO) algorithm appears among heuristic algorithms used for solving discrete optimization problems. In this study, a new hybrid method is proposed to optimize parameters that affect performance of the ACO algorithm using Particle Swarm Optimization (PSO). In addition, 3-Opt heuristic method is added to proposed method in order to improve local solutions. The PSO algorithm is used for detecting optimum values of parameters α and β which are used for city selection operations in the ACO algorithm and determines significance of inter-city pheromone and distances. The 3-Opt algorithm is used for the purpose of improving city selection operations, which could not be improved due to falling in local minimums by the ACO algorithm. The performance of proposed hybrid method is investigated on ten different benchmark problems taken from literature and it is compared to the performance of some well-known algorithms. Experimental results show that the performance of proposed method by using fewer ants than the number of cities for the TSPs is better than the performance of compared methods in most cases in terms of solution quality and robustness.

1. Introduction

The Traveling Salesman Problem (TSP) is a well-known combinatorial discrete optimization problem where the salesman attempts to find the shortest tour through cities. This problem has been used in many engineering applications such as the design of hardware devices and radio electronic systems, and computer networks [1,2].

In the theory of computational complexity, the decision version of the TSP (where, given a length L, the task is to decide whether the graph has any tour shorter than L) belongs to the class of NP-complete problems. Thus, it is possible that the worst-case running time for any algorithm for the TSP increases exponentially with the number of cities [2]. For this reason, in recent years some heuristic algorithms have been proposed to solve this problem, which have achieved better results in terms of computational and time complexity. Grefenstette et al. presented some approaches to the application of Genetic Algorithms (GA) to the TSP [3]. Shi et al. presented a Particle Swarm Optimization (PSO) based algorithm for the TSP [4]. Geng et al. proposed an effective local search algorithm based on Simulated Annealing (SA) and greedy search techniques to solve the TSP [5]. In order to obtain more accuracy solutions, the proposed algorithm based on the standard SA algorithm adopted the combination of three kinds of mutations with different probabilities during its search. Jolai and Ghanbari presented an improved Artificial Neural Network (ANN) approach for solving the TSP [6]. They employed Hopfield Neural Network and data transformation techniques together to improve accuracy of the results and reach to the optimal tours with less total distances. Pedro et al. proposed a Tabu Search algorithm to solve the TSP [7]. Dorigo et al. proposed an Ant System to solve the TSP [8]. Dorigo and Gambardella described an artificial ant colony (ACO) capable of solving the TSP [9]. They demonstrated that the ACO was capable of generating good solutions to both symmetric and asymmetric instances of the TSP. Mavrovouniotis and Yang proposed an ACO framework for dynamic environments [10]. Their framework contains different immigrants schemes, including random immigrants, elitisim-based immigrants, and memory based immigrants. Karaboga and Gorkemli proposed a new Artificial Bee Colony (ABC) algorithm called Combinatorial ABC for the TSP [11]. They showed that the ABC algorithm can be used for combinatorial optimization problems.

* Corresponding author. Tel.: +90 3322233711; fax: +90 3322410635.
E-mail address: hkodaz@gmail.com (H. Kodaz).

http://dx.doi.org/10.1016/j.asoc.2015.01.068
1568-4946/

To solve this problem, hybrid heuristic methods based on Simulated Annealing, PSO, ACO, ABC, ANN, etc. were used. Bountoux and Feillet proposed a hybrid algorithm to solve the TSP [12]. Their algorithm consists of the ACO algorithm hybridized with local search procedures. They called Dynamic Multi-Dimensional Anamorphic Travelling Ants (DMD-ATA). Tsai et al. presented a metaheuristic approach called ACOMAC algorithm for solving the TSP [13]. They introduced multiple ant clans' concept from parallel genetic algorithm to search solution space utilizing various islands to avoid local minima and thus can yield global minimum for solving the TSP. Also, they presented two approaches named the multiple nearest neighbor (NN) and the dual nearest neighbor (DNN) to ACOMAC to enhance large TSPs. Pasti and Castro proposed a meta-heuristics for solving the TSP based on a neural network trained using ideas from the immune system [14]. The network was self-organized and the learning algorithm aims at locating one network cell at each position of a city of the TSP instance to be solved. Their network based on a Real-valued Antibody Network (RABNET). Masutti and Castro proposed some modifications on the RABNET-TSP, an immune-inspired self-organizing neural network, for the solution of the TSP [15]. Beam-ACO algorithm, which is a hybrid method combining ACO with beam search was used to solve TSP [16]. Cheng and Mao developed a modified ant algorithm, named Ant Colony System-Traveling Salesman Problem with Time Windows (ACS-TSPTW), based on the ACO technique to solve the TSP [17]. Krohling and Coelho presented an approach based on co-evolutionary PSO for solving the constrained optimization problems as min–max problems [18]. Lin et al. presented an evolutionary neural fuzzy network, designed using the functional-link-based neural fuzzy network and an evolutionary learning algorithm [19]. Their evolutionary learning algorithm was based on a hybrid cooperative PSO and cultural algorithms for prediction problems. Chen and Chien presented a method, called the genetic simulated annealing ant colony system with Particle Swarm Optimization techniques, for solving the TSP [20]. Junqiang and Aijia proposed a Hybrid Ant Colony Algorithm (HACO) which is containing ACO and delete-cross method which is used to speed the convergence of local search is presented for the shortcoming that the convergence speed of ACO is a bit slow [21]. Dong et al. presented an approach, called Cooperative Genetic Ant System (CGAS), combines both GA and ACO together in a cooperative manner to improve the performance of ACO for solving TSP [22]. Peker et al. proposed for the TSP using the ant colony system and parameter optimization was taken from the Taguchi method [23]. Gunduz and Kiran presented a new hierarchic method based swarm intelligence algorithms for solving well-known TSP [24]. The swarm intelligence algorithms implemented in their study were divided into two types as path construction and path improvement based methods. The path construction based method (Ant Colony Optimization-ACO) that produced good solutions have taken more time to achieve a good solution and also, the path improvement based technique (Artificial Bee Colony – ABC) that quickly produced results have not achieved a good solution in a reasonable time. Therefore, their hierarchic method which consists of ACO-ABC was proposed to achieve a good solution in a reasonable time. ACO was used to provide better initial solution for ABC that use path improvement technique in order to achieve to optimal or near optimal solution.

In this study, a new hybrid method was suggested, which optimizes parameters that affect performance of the ACO algorithm through PSO and reduces the probability of falling in local minimum with the 3-Opt algorithm. Better results were achieved with the suggested method compared to other studies in the literature by using fewer ants than the number of cities for the TSPs.

The rest of the paper is organized as follows: In Section 2, Materials and methods gives the background information including PSO, ACO and 3-Opt algorithms. The proposed method is presented in Section 3. The results obtained in the application are given in Section 4. Consequently, in Section 5, we conclude the paper with a summarization of results by emphasizing the importance of this study.

2. Materials and methods

In this section, information is provided on the PSO, ACO and 3-Opt algorithms.

2.1. Particle Swarm Optimization (PSO) algorithm

PSO is a population-based optimization algorithm developed by Kennedy and Eberhart and inspired by bird flocks' behavior when searching for food [25]. Each individual in the swarm is called a particle and points to a solution in search space. Particles have dimensions similar to the number of parameters whose values are desired to be found in a problem. Particles are randomly distributed in search space at first. Each particle generates a fitness value depending on the objective function of the problem. Locations of particles are updated as in Eqs. (1) and (2) [25].

$$v_i^{t+1} = v_i^t + c_1 r_1^t (Pbest_i^t - X_i^t) + c_2 r_2^t (gbest - X_i^t) \tag{1}$$

$$X_i^{t+1} = X_i^t + v_i^{t+1} \tag{2}$$

where X_i^t indicates ith particle's location in t iteration, X_i^{t+1} indicates ith particle's location in $t+1$ iteration and v_i^{t+1} indicates velocity vector of ith particle. c_1 and c_2 determine impact of the particle's own best solution ($Pbest_i$) and the system's best solution ($gbest$) on velocity vector and r_1 and r_2 are random numbers at interval [0–1]. In improved versions of the PSO, the parameter of inertia weight w, which determines impact of old velocity vector on new velocity vector, was added to Eq. (1) [25]. Algorithm is operated until the designated number of iterations or error value has been achieved.

2.2. Ant Colony Optimization (ACO) algorithm

The ACO algorithm was developed by Dorigo et al. as inspired by actual ant colony behaviors [8]. Having examined behaviors of ants in real life, it is observed that ants have the ability to find the shortest route between their nest and food source. The most important feature in finding the shortest route is the volatile, chemical matter of pheromone that ants leave on the path they use. Ants in a colony generally choose a path where pheromone matter is concentrated. The amount of pheromone increases on a frequently used route [26]. The algorithm that suggests a solution to the TSP, which is a discrete (combinatorial) test problem, by utilizing this attribute of ants was proposed by Dorigo et al. [27]. In the TSPs, the traveling salesman aims to form a closed tour of minimum length provided that it visits every city once. In this proposed method, it is accepted that ants leave pheromones on inter-city routes that they use and this pheromone becomes volatile in a certain ration. Selection of the cities to which ants will go is greedily performed depending on the distance and the amount of pheromones between cities. This algorithm is operated iteratively and the shortest route found is regarded as the best solution. Selection of city j, to which an ant in city i in iteration t will go, is made according to Eq. (3).

$$P_{ij}{}^{k} = \begin{cases} \dfrac{[\tau_{ij}(t)]^{\alpha}[\eta_{ij}]^{\beta}}{\sum [\tau_{ij}(t)]^{\alpha}[\eta_{ij}]^{\beta}}, & \text{if } j \text{ is allowed city} \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

In Eq. (3), τ_{ij} indicates the amount of pheromones between i and j cities, η_{ij} indicates information ($1/d_{ij}$) pertaining to distance between i and j cities, and j display cities where kth ant can go. An

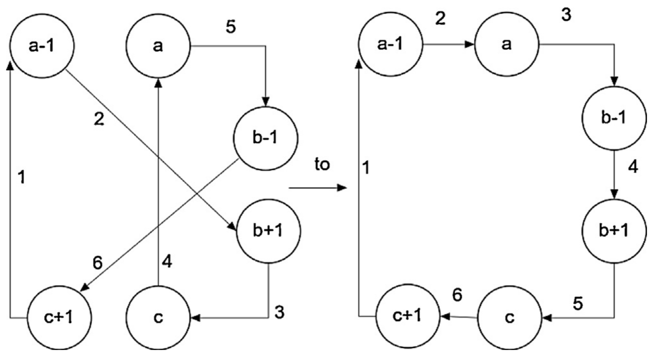

Fig. 1. 3-Opt algorithm representation [26].

-Initialization (number of ant, number of city, pheromone, iteration number)
-First tour length calculation using nearest city
-Calculation of new pheromone values
-Optimization of ACO parameters (α, β) with PSO
While (ant ≤ number of ant)
Initialize parameters for PSO,(particle=number of ant, c_1,c_2)
While (t<iteration number)
While (particle ≤ number of ant)

$$p_{ij}^{k}=\begin{cases}\frac{\tau_{ij}^{\alpha}\eta_{ij}^{\beta}}{\sum_{r\epsilon\Gamma}^{n}\tau_{ir}^{\alpha}\eta_{ir}^{\beta}}, if j \notin \Gamma \\ 0, otherwise\end{cases}$$

Send parameters to ACO from PSO find the besttour
Calculation new α and β values
$V_i(k+1) = wV_i(k) + c_1r_1\ Pi - x_i(k) + c_2r_2Pg - x_i(k)$,
$X_i(k+1) = X_i(k) + V_i(k+1)$,
- Update pheromone values
- Execution algorithm 3-OPT

Fig. 2. Pseudo-code of proposed method.

ant chooses the city with the highest ratio of P_{ij} by making a greedy selection. Parameters α vs β are used for determining the significance between amount of pheromones and distance inter-city. kth ant completes one total tour by using Eq. (3). The abovementioned operation is repeated in t iteration for all ants that are present in the colony. The amount of pheromones left by an ant on a route that it has used is determined according to Eq. (4).

$$\Delta\tau_{ij}^{k}(t,t+1)=\begin{cases}\frac{Q}{L^k}, & \text{if } (i,j)\in \text{route performed by the } k\text{th ant} \\ 0, & \text{otherwise}\end{cases} \quad (4)$$

where L^k represents distance of tour, Q represents a constant number and k represents kth ant in the colony. Total amount of pheromones that ants, which are present in colony and use the route between cities i and j have left, is calculated by using Eq. (5).

$$\Delta\tau_{ij}^{k}(t,t+1)=\sum_{k=1}^{n}\Delta\tau_{ij}^{k}(t,t+1) \quad (5)$$

Amount of pheromones, which will be found in inter-city routes in iteration (t+1), is determined as in Eq. (6) depending on the impact of evaporation as well.

$$\tau_{ij}(t+1)=(1-\rho)\tau_{ij}(t)+\Delta\tau_{ij}^{k}(t,t+1) \quad (6)$$

In Eq. (6), ρ is the coefficient of evaporation and receives a value at intervals [0–1]. When the maximum number of iterations is reached, the shortest tour length obtained is regarded as the solution of the problem.

2.3. 3-Opt algorithm

3-Opt is a simple local search algorithm for solving the TSP in optimization [28]. 3-Opt algorithm is a special case of the k-opt algorithm [26]. In this algorithm, 3-Opt analysis involves deleting three connections (or edges) in a network (or tour), reconnecting the network in all other possible ways, and then evaluating each reconnection method to find the optimum one. This process is then repeated for a different set of three connections [26]. In this way, there are edges in the graph as shown in Fig. 1 and edges overlapped in the tour are created on the graph. This leads to increasing lengths of tour. Using replacements for three edges in the specified nodes determines the length of the best tour as illustrated in Fig. 1.

To reduce the length of the best tour, different algorithms such as GA, PSO, ACO and ABC are presented. When these algorithms try to find the best tour, they fall in local minimum and this leads to a lack of obtain lengths of the best tour. To eliminate local minimum situations, they have been used in a k-opt algorithm in [26] and one of them is a 3-Opt algorithm.

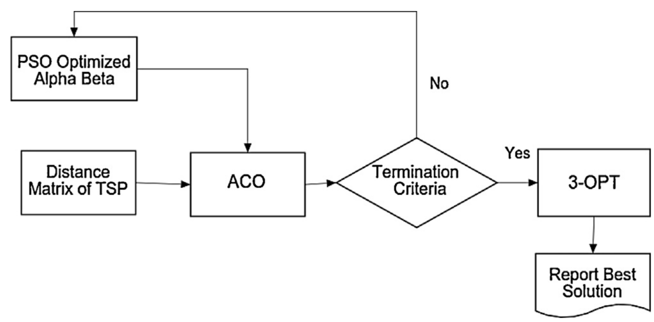

Fig. 3. The flowchart of the proposed method.

3. Proposed method (PSO–ACO–3Opt)

In general, the number of ants is taken as equal to the number of cities for the solution of the TSPs via ACO. Increasing the number of ants also increases the calculation complexity. Also, parameters α and β are determined according to experience. In this study, a hybrid method is proposed that is based on the PSO, ACO and 3-Opt algorithms in order to improve solution performance of the TSPs. At first, ants are randomly distributed to cities. Then, pheromones are assigned to all inter-city routes as much as the amount calculated by the formula in Eq. (7).

$$\text{Amount of pheromone}=\frac{1}{\text{number of ant} * \text{number of city}} \quad (7)$$

All ants complete their first tours only by taking inter-city distances into account. Tour lengths are determined for all ants and the pheromone update is realized according to Eqs. (4)–(6). Values of parameters α and β in Eq. (3) are determined by using the PSO. Objective function of the PSO algorithm is the tour length. $gbest_{ant}$ represents parameters α and β, which yield the shortest tour length for each ant in the PSO algorithm. Ant route and parameters that provide the shortest tour length are accepted as the solution of the system. Pheromone update is achieved according to Eq. (6) by using routes of all ants. When the number of iterations designated for the ACO algorithm is reached, the stage of the PSO–ACO has been completed. The 3-Opt algorithm is applied after the stage of the PSO–ACO for not falling in local optimum. In our proposed method, a solution is developed by applying the 3-Opt algorithm to the best solution. Pseudo-code, which belongs to the proposed method, is shown in Fig. 2 and the flowchart of the system is shown in Fig. 3.

4. Experimental results

Performance of the proposed method for the TSPs was attempted to be determined by using standard deviation values and average tour length on ten different test problems taken from

Table 1
The effects of ant numbers on the performance. Avg is the average route length; SD is the standard deviation; error (%) is percentage relative error; time (s) is run time in seconds.

Data set		Ant number = 10	Ant number = 20	Ant number = 30	Ant number = city number
Eil51	Avg.	**426.45**	427.50	429.25	432.75
	SD	0.61	0.53	1.16	1.49
	Error (%)	0.11	0.35	0.76	1.58
	Time (s)	140.50	141.29	149.52	160.05
Berlin52	Avg.	**7543.20**	7580.38	7586.63	7598.63
	SD	2.37	22.03	22.93	30.98
	Error (%)	0.02	0.51	0.59	0.75
	Time (s)	170.46	173.32	174.10	180.90
Rat99	Avg.	**1227.40**	1240.37	1251.38	1254.13
	SD	1.98	6.41	6.00	7.06
	Error (%)	0.28	1.34	2.24	2.46
	Time (s)	284.09	294.77	305.98	326.58
Eil76	Avg.	**538.30**	540.38	546.75	549.73
	SD	0.47	1.77	1.98	2.87
	Error (%)	0.06	0.44	1.63	2.18
	Time (s)	220.68	239.95	279.17	283.65
St70	Avg.	**678.20**	680.875	681.56	683.50
	SD	1.47	1.73	3.13	1.77
	Error (%)	0.47	0.87	0.97	1.26
	Time (s)	256.89	260.15	273.20	291.61
Kroa100	Avg.	**21,445.10**	21,709.63	21,816.44	21,974.00
	SD	78.24	113.11	218.15	115.88
	Error (%)	0.77	2.01	2.51	3.25
	Time (s)	301.32	303.06	310.17	332.40
Lin105	Avg.	**14,379.15**	14,593.75	14,664.88	14,690.63
	SD	0.48	66.04	58.36	72.28
	Error (%)	0.00	1.49	1.99	2.17
	Time (s)	294.35	303.24	310.29	349.16
Kroa200	Avg.	**29,646.05**	30,357.63	30,504.25	30,662.75
	SD	114.71	51.00	148.50	202.56
	Error (%)	0.95	3.37	3.44	4.41
	Time (s)	302.15	308.43	380.42	1179.46
CH150	Avg.	**6563.95**	6727.25	6779.13	6800.13
	SD	27.58	32.69	7.55	23.93
	Error (%)	0.55	3.05	3.85	4.17
	Time (s)	286.90	300.93	329.83	346.37
Eil101	Avg.	**632.70**	646.00	647.25	647.75
	SD	2.12	1.77	2.49	1.83
	Error (%)	0.59	2.70	2.90	2.98
	Time (s)	302.15	302.32	302.70	330.40

TSPLIB [29]. Both number of ants and particles are selected as 10 in all experiments. The effects of different number of ants on the performance are shown in Table 1 for the Eil51, Berlin52, Rat99, Eil76, St70, Kroa100, Lin105, Kroa200, Ch150 and Eil101 test problems, respectively. As seen from Table 1, run time for the application is increased and performance is getting worse according to number of ants. The best results are given in bold.

The best values of α, β are selected at interval of $0 \geq \alpha \geq \beta \geq 2$. Every particle has two dimensions as (α, β) in the PSO algorithm. For the PSO, parameters c_1 and c_2 are selected as 2. The best parameter values for the ACO are given in Table 2. Both ACO and PSO algorithms are operated throughout 1000 iterations. Tests for each test problem are repeated 20 times. Pheromone evaporation ratio for the ACO algorithm is selected as 0.1. Value of evaporation ratio is determined in a way that will yield the best value after trial and error.

Graphics of the best solutions obtained from applied and unapplied the 3-Opt algorithm for test problems are shown in Fig. 4. Also lengths of the route obtained from applied and unapplied 3-Opt algorithm are given in Fig. 4. As can be seen from Fig. 4, encircled tour segments could not be efficiently found by the PSO–ACO. After using the 3-Opt algorithm, more efficient tours are achieved.

Table 2
The best parameters values for ACO.

Problem	α	β
Eil51	1.11	1.44
Berlin52	0.95	1.05
Rat99	0.99	1.07
Eil76	0.88	1.50
St70	0.94	1.05
Kroa100	1.01	1.10
Lin105	1.20	0.65
Kroa200	0.75	1.15
Ch150	0.75	1.20
Eil101	1.20	0.75

Performance of this proposed method is assessed by running it 20 times and calculated the average value, standard deviation value and percentage relative error according to optimal solution. The percentage relative error is defined as Eq. (8) [20]. The results obtained by proposed method are given in Table 3. Results of this assessment are provided in Table 4 in a comparative way along with other studies in the literature. The best results are given in bold.

$$\text{Error}\,(\%) = \frac{(\text{average solution} - \text{best known solution})}{\text{best known solution}} \times 100 \qquad (8)$$

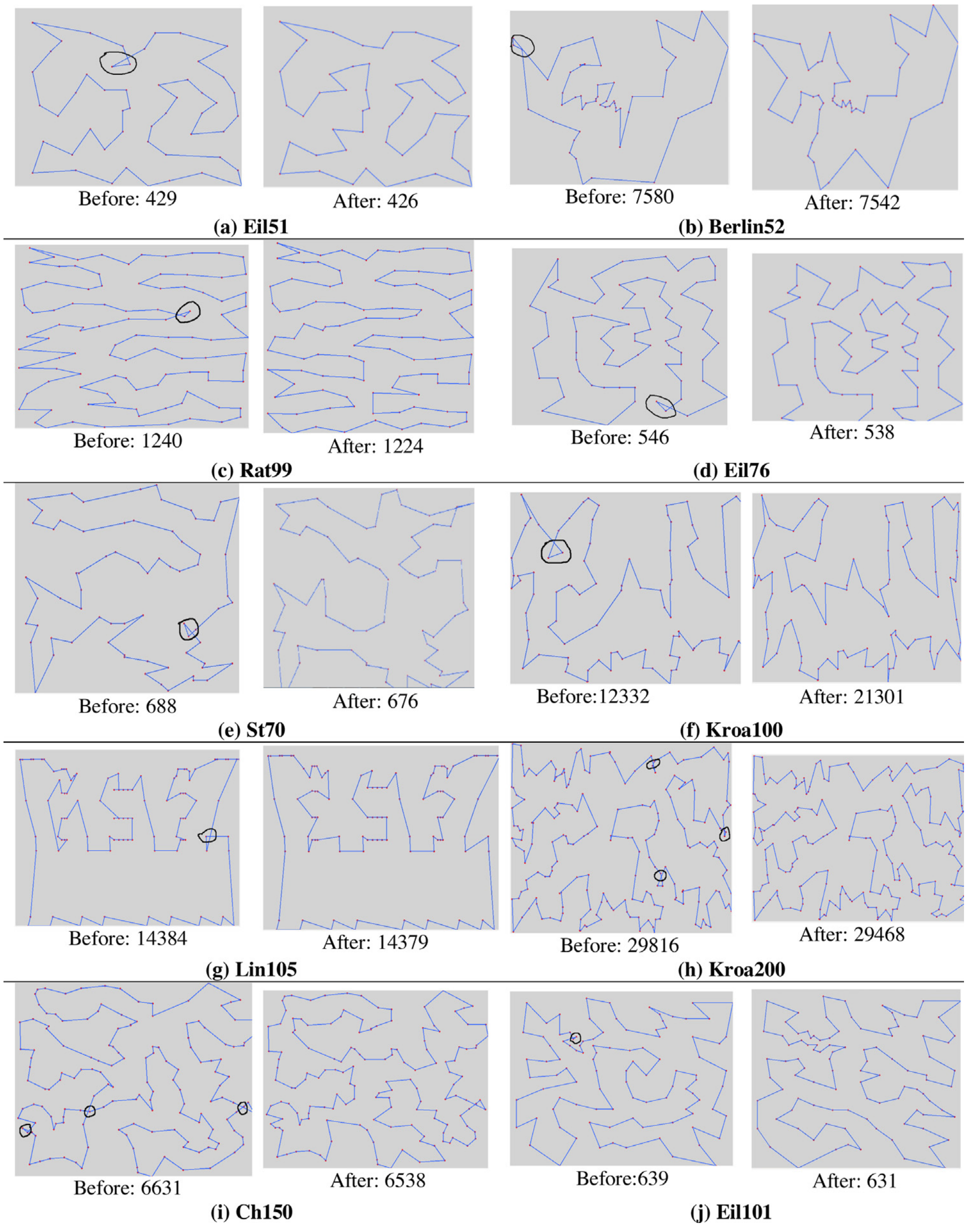

Fig. 4. Best routes found before and after execution of 3-Opt by the proposed method.

Examining Table 4, it can be seen that the proposed method has generated the closest results to optimal solution with minimal standard deviation for problems the Eil51, Rat99, Eil76, St70, Kroa200 and Eil101. The average results obtained for the Eil51, Rat99, Eil76, St70, Kroa200 and Eil101 are 426.45, 1227.40, 538.30, 678.20, 29,646.05 and 632.70, respectively. As will be seen from Table 4, these results are better than the results of studies in the literature. It is also observed that results close to optimal solution

Table 3
The results obtained by the proposed method for test problems.

Problem	BKS[a]	Best	Worst	Average	SD[b]	Error (%)[c]	Time (s)
Eil51	426	426	428	426.45	0.61	0.11	140.50
Berlin52	7542	7542	7548	7543.20	2.37	0.02	170.46
Rat99	1224	1224	1230	1227.40	1.98	0.28	284.09
Eil76	538	538	539	538.30	0.47	0.06	220.68
St70	675	676	681	678.20	1.47	0.47	256.89
Kroa100	21,282	21,301	21,554	21,445.10	78.24	0.77	301.32
Lin105	14,379	14,379	14,381	14,379.15	0.48	0.00	294.35
Kroa200	29,368	29,468	29,957	29,646.05	114.71	0.95	303.23
Ch150	6528	6538	6622	6563.95	27.58	0.55	286.90
Eil101	629	631	638	632.70	2.12	0.59	302.15

[a] Best known solution.
[b] Standard deviation.
[c] Percentage relative error for the results obtained by 20 runs.

Table 4
The computational results of the proposed method and other methods in the literature. BKS is the best known solution; Avg is the average route length; SD is the standard deviation; error (%) is percentage relative error.

Method	Problem	Eil51	Berlin52	Rat99	Eil76	St70	Kroa100	Lin105	Kroa200	Ch150	Eil101
	BKS	426	7542	1224	538	675	21,282	14,379	29,368	6528	629
ACOMAC (2004) [13]	Avg.	430.68	–	–	555.70	–	21,457.00	–	–	–	–
	SD	–	–	–	–	–	–	–	–	–	–
	Error (%)	1.10	–	–	3.29	–	0.82	–	–	–	–
ACOMAC + NN (2004) [13]	Avg.	430.68	–	–	555.90	–	21,433.30	–	–	–	–
	SD	–	–	–	–	–	–	–	–	–	–
	Error (%)	1.10	–	–	3.33	–	0.71	–	–	–	–
RABNET – TSP (2006) [14]	Avg.	438.70	8073.97	–	556.10	–	21,868.47	14,702.17	30,257.53	6753.20	654.83
	SD	3.52	270.14	–	8.03	–	245.76	328.37	342.98	83.01	6.57
	Error (%)	2.98	7.05	–	3.36	–	2.76	2.25	3.03	3.45	4.11
Modified RABNET – TSP (2009) [15]	Avg.	437.47	7932.50	–	556.33	–	21,522.73	14,400.7	30,190.27	6738.37	648.63
	SD	4.20	277.25	–	5.30	–	93.34	44.03	273.38	76.14	3.85
	Error (%)	2.69	5.18	–	3.41	–	1.13	0.15	2.80	3.22	3.12
SA ACO PSO (2011) [20]	Avg.	427.27	**7542.00**	–	540.20	–	21,370.30	14,406.37	29,738.73	**6563.70**	635.23
	SD	0.45	0.00	–	2.94	–	123.36	37.28	356.07	22.45	3.59
	Error (%)	0.30	0.00	–	0.41	–	0.41	0.19	1.26	0.55	0.99
IVRS + 2opt (2012) [30]	Avg.	431.10	7547.23	–	–	–	21,498.61	–	–	–	648.67
	SD	–	–	–	–	–	–	–	–	–	–
	Error (%)	1.20	0.07	–	–	–	1.02	–	–	–	3.13
ACO + 2opt (2012) [30]	Avg.	439.25	7556.58	–	–	–	23,441.80	–	–	–	672.37
	SD	–	–	–	–	–	–	–	–	–	–
	Error (%)	3.11	0.19	–	–	–	10.15	–	–	–	6.90
HACO (2012) [21]	Avg.	431.20	7560.54	1241.33	–	–	–	–	–	–	–
	SD	2.00	67.48	9.60	–	–	–	–	–	–	–
	Error (%)	1.22	0.23	1.42	–	–	–	–	–	–	–
CGAS (2012) [22]	Avg.	–	7634.00	–	542.00	–	21,437.00	–	29,946.00	–	–
	SD	–	–	–	–	–	–	–	–	–	–
	Error (%)	–	1.22	–	0.74	–	0.73	–	1.97	–	–
WFA with 2-Opt (2013) [31]	Avg.	426.65	**7542.00**	–	541.22	–	**21,282.00**	**14,379.00**	29,654.03	6572.13	639.87
	SD	0.66	0.00	–	0.66	–	0.00	0.00	151.42	13.84	2.88
	Error (%)	0.15	0.00	–	0.60	–	0.00	0.00	0.97	0.68	1.73
WFA with 3-Opt (2013) [31]	Avg.	426.60	**7542.00**	–	539.44	–	21,282.80	14,459.40	29,646.50	6700.10	633.50
	SD	0.52	0.00	–	1.51	–	0.00	1.38	110.91	60.82	3.47
	Error (%)	0.14	0.00	–	0.27	–	0.00	0.56	0.95	2.64	0.72
ACO with Tagushi Method (2013) [23]	Avg.	435.40	7635.40		565.50		21,567.10	14,475.20			655.00
	SD	–	–	–	–	–	–	–	–	–	–
	Error (%)	2.21	1.24	–	5.11	–	1.34	0.67	–	–	4.13
ACO with ABC (2014) [24]	Avg.	443.39	7544.37	–	557.98	700.58	22,435.31	–	–	6677.12	683.39
	SD	5.25	0.00	–	4.10	7.51	231.34	–	–	19.30	6.56
	Error (%)	4.08	0.03	–	3.71	3.79	5.42	–	–	2.28	8.65
Proposed Method PSO–ACO–3Opt	Avg.	**426.45**	7543.20	**1227.40**	**538.30**	**678.20**	21,445.10	14,379.15	**29,646.05**	6563.95	**632.70**
	SD	0.61	2.37	1.98	0.47	1.47	78.24	0.48	114.71	27.58	2.12
	Error (%)	0.11	0.02	0.28	0.06	0.47	0.77	0.00	0.95	0.55	0.59

have been obtained for the Berlin52, Lin105 and Ch150, and also reasonable results have been obtained in the Kroa100. For example, while it was obtained 6563.70 for the Ch150 problem in the literature, proposed method found 6563.95. We obtained the least percentage relative error (0.55%) for the Ch150. Examining results in the literature, it is seen that this proposed method has produced closer results to optimum.

5. Conclusion

In this study, a new hybrid method based on the PSO, ACO and 3-Opt algorithms is proposed in order to solve the TSPs. In this method, the PSO is used for determining parameters α and β which affected performance of the ACO, and the 3-Opt is used for getting rid of the local solution found the ACO algorithm. The performance of this proposed method is investigated by taking into consideration average route length, standard deviation and percentage relative error values according to average value on ten different test problems taken from TSPLIB. The effects of the different number of ants in the ACO are also analyzed in this present study. As seen from the experimental results, performance of the proposed method is getting better depended on the fewer number of ants. From the results obtained in this work, it can be concluded that the performance of the proposed method is better than or similar to performance of compared methods.

References

[1] G. Laporte, The Traveling Salesman Problem – an overview of exact and approximate algorithms, Eur. J. Oper. Res. 59 (1992) 231–247.
[2] Wikipedia, Travelling Salesman Problem, http://en.wikipedia.org/wiki/Travelling_salesman_problem
[3] J. Grefenstette, R. Gopal, B. Rosmaita, D. Van Gucht, Genetic algorithms for the traveling salesman problem, in: Proceedings of the First International Conference on Genetic Algorithms and their Applications, Lawrence Erlbaum, NJ, 1985, pp. 160–168.
[4] X.H. Shi, Y.C. Liang, H.P. Lee, C. Lu, Q.X. Wang, Particle swarm optimization-based algorithms for TSP and generalized TSP, Inf. Process. Lett. 103 (2007) 169–176.
[5] X.T. Geng, Z.H. Chen, W. Yang, D.Q. Shi, K. Zhao, Solving the traveling salesman problem based on an adaptive simulated annealing algorithm with greedy search, Appl. Soft Comput. 11 (2011) 3680–3689.
[6] F. Jolai, A. Ghanbari, Integrating data transformation techniques with Hopfield neural networks for solving travelling salesman problem, Expert Syst. Appl. 37 (2010) 5331–5335.
[7] O. Pedro, R. Saldanha, R. Camargo, A Tabu Search Approach for the prize collecting Traveling Salesman Problem, Electron. Notes Discrete Math. 41 (2013) 261–268.
[8] M. Dorigo, V. Maniezzo, A. Colorni, Ant system: optimization by a colony of cooperating agents, IEEE Trans. Syst. Man Cybern. B 26 (1996) 29–41.
[9] M. Dorigo, L.M. Gambardella, Ant colonies for the travelling salesman problem, Biosystems 43 (1997) 73–81.
[10] M. Mavrovouniotis, S.X. Yang, Ant colony optimization with immigrants schemes for the dynamic travelling salesman problem with traffic factors, Appl. Soft Comput. 13 (2013) 4023–4037.
[11] D. Karaboga, B. Gorkemli, A combinatorial artificial bee colony algorithm for traveling salesman problem, in: 2011 International Symposium on Innovations in Intelligent Systems and Applications (INISTA), IEEE, 2011, pp. 50–53.
[12] B. Bontoux, D. Feillet, Ant colony optimization for the traveling purchaser problem, Comput. Oper. Res. 35 (2008) 628–637.
[13] C.F. Tsai, C.W. Tsai, C.C. Tseng, A new hybrid heuristic approach for solving large traveling salesman problem, Inf. Sci. 166 (2004) 67–81.
[14] R. Pasti, L.N. De Castro, A neuro-immune network for solving the traveling salesman problem, in: International Joint Conference on Neural Networks, 2006. IJCNN'06, IEEE, 2006, pp. 3760–3766.
[15] T.A.S. Masutti, L.N. de Castro, A self-organizing neural network using ideas from the immune system to solve the traveling salesman problem, Inf. Sci. 179 (2009) 1454–1468.
[16] M. Lopez-Ibanez, C. Blum, Beam-ACO for the travelling salesman problem with time windows, Comput. Oper. Res. 37 (2010) 1570–1583.
[17] C.B. Cheng, C.P. Mao, A modified ant colony system for solving the travelling salesman problem with time windows, Math. Comput. Model. 46 (2007) 1225–1235.
[18] R.A. Krohling, L.D. Coelho, Coevolutionary particle swarm optimization using Gaussian distribution for solving constrained optimization problems, IEEE Trans. Syst. Man Cybern. B 36 (2006) 1407–1416.
[19] C.J. Lin, C.H. Chen, C.T. Lin, A hybrid of cooperative Particle Swarm Optimization and cultural algorithm for neural fuzzy networks and its prediction applications, IEEE Trans. Syst. Man Cybern. C 39 (2009) 55–68.
[20] S.M. Chen, C.Y. Chien, Solving the traveling salesman problem based on the genetic simulated annealing ant colony system with particle swarm optimization techniques, Expert Syst. Appl. 38 (2011) 14439–14450.
[21] W. Junqiang, O. Aijia, A hybrid algorithm of ACO and delete-cross method for TSP, in: 2012 International Conference on Industrial Control and Electronics Engineering (ICICEE), IEEE, 2012, pp. 1694–1696.
[22] G.F. Dong, W.W. Guo, K. Tickle, Solving the traveling salesman problem using cooperative genetic ant systems, Expert Syst. Appl. 39 (2012) 5006–5011.
[23] M. Peker, B. Sen, P.Y. Kumru, An efficient solving of the traveling salesman problem: the ant colony system having parameters optimized by the Taguchi method, Turk. J. Electr. Eng. Comput. 21 (2013) 2015–2036.
[24] M. Gunduz, M.S. Kiran, E. Ozceylan, A hierarchic approach based on swarm intelligence to solve traveling salesman problem, Turk. J. Electr. Eng. Comput. Sci. (2014), http://dx.doi.org/10.3906/elk-1210-147.
[25] J. Kennedy, R. Eberhart, Particle swarm optimization, in: 1995 IEEE International Conference on Neural Networks Proceedings, vols. 1–6, 1995, pp. 1942–1948.
[26] M. Dorigo, T. Stutzle, Ant Colony Optimization, Massachusetts Institute of Technology, 2004.
[27] M. Dorigo, V. Maniezzo, A. Colorni, V. Maniezzo, Positive Feedback as a Search Strategy, 1991.
[28] Wikipedia, 3-Opt Algorithm, http://en.wikipedia.org/wiki/3-opt
[29] G. Reinelt, TSPLIB—a traveling salesman problem library, ORSA J. Comput. 3 (1991) 376–384.
[30] K. Jun-man, Z. Yi, Application of an improved Ant Colony Optimization on generalized Traveling Salesman Problem, Energy Proc. 17 (2012) 319–325.
[31] Z.A. Othman, A.I. Srour, A.R. Hamdan, P.Y. Ling, Performance water flow-like algorithm for TSP by improving its local search, Int. J. Adv. Comput. Technol. 5 (2013).