Paper 10

by The Jin Ai

Submission date: 19-Jul-2019 02:41PM (UTC+0700)
Submission ID: 1153159895
File name: Paper_10_ASOR_SI_for_TSP.pdf (2.28M)
Word count: 100
Character count: 530

ASOR Bulletin
Volume 30, Number 4, December 2011, pp.2-10
ISSN0812-860X | eISSN1446-6678

A Swarm-Intelligence Based Algorithm for Solving Traveling Salesman Problem

The Jin Ai
Department of Industrial Engineering
Universitas Atma Jaya Yogyakarta, INDONESIA
Email: jinai@mail.uajy.ac.id

Abstract

The traveling salesman problem (TSP) plays an important role in the field of physical distribution and logistics. This problem is generally defined as the problem for determining the sequence of the cities to be visited by a traveling salesman such that the operational cost of the traveling is minimized. In recent two decades, the swarm-intelligence philosophy has been emerging into some optimization algorithms such as ant colony optimization (ACO) and its variants, also particle swarm optimization (PSO) and its variants. Both ACO and PSO, despite their strengths and weaknesses, are reported successfully being applied to solve the TSP. This paper tries to develop a new solution methodology for solving the traveling salesman problem using different point of view on the swarm-intelligence philosophy in the form of a new proposed algorithm. The expectations of the new proposed algorithm are reducing the weakness and increasing the strength of its application. The performance of the proposed algorithm is then evaluated using several benchmark datasets for the traveling salesman problem. The computational results show that the proposed algorithm using specific settings is able to find good solution of the corresponding traveling salesman problem instance. It is also still possible to improve the result by fine tuning the algorithm and adding an effective local search method.

Key Words: Traveling Salesman Problem, Swarm Intelligence, Algorithm, Metaheuristics

1. Introduction

The traveling salesman problem (TSP), which has an important role in the field of physical distribution and logistics, is generally defined as the problem for determining the sequence of the cities to be visited by a traveling salesman such that the operational cost or the distance of the traveling is minimized (Laporte, 2010). The TSP can be formally defined as follows. Let $G=(V,A)$ be a graph where $V=\{v_1,\ldots,v_n\}$ is a vertex set, and $A=\{(v_i,v_j)|v_i,v_j\in V, i\neq j\}$ is an edge set. Each vertex represents city to be visited by a travelling salesman. Associated with A are a nonnegative cost or distance matrix (c_{ij}).The TSP consists of finding a Hamiltonian cycle of G, a tour that passes through all the vertices, with minimum cost. Since TSP is a hard optimization problem, many heuristics and metaheuristics have been proposed for solving this problem (Laporte, 2010; Rego et al., 2011).

In recent two decades, the swarm-intelligence philosophy has been emerging into some optimization algorithms such as ant colony optimization (ACO) and its variants (Bonabeau, 1999), also particle swarm optimization (PSO) and its variants (Kennedy and Eberhart, 2001; Clerc, 2006). Despite their strengths and weaknesses, both ACO and PSO are reported successfully being applied to solve the TSP (Dorigo and Gambardella, 1997; Chen and Chien, 2011; Zhong et al., 2007; Tasgetiren et al., 2007). It is noted that both ACO and PSO are able to generate number of potential salesman tours in their iteration step, whereas the best tour found is improved by their mechanism from iteration to iteration. In general, the performance of ACO and PSO algorithms heavily depend on the solution representation, mechanism, and parameter used in the algorithms. In some PSO application for solving TSP, the tour representation required complicated transformation to be converted into salesman tour. In other applications which use simple representation, it is often observed that the PSO mechanism is complex in order to maintain feasibility of the tour. Both complex transformation and complex mechanism usually consumes significant computational effort, hence, reduce the algorithm speed.

This paper tries to develop a new solution methodology for solving the traveling salesman problem using different point of view on the swarm-intelligence philosophy in the form of a new proposed algorithm. The PSO is selected as the reference algorithm for developing new algorithm. Instead of indirect representation, direct representation of salesman tour is employed in the algorithm for eliminating the transformation process.

2. Development of the Swarm-Intelligence Based Algorithm

The development of the swarm-intelligence based algorithm for solving the traveling salesman problem is starting with the identification of the main characteristics of the particle swarm optimization (PSO), the selected swarm-intelligence algorithm as reference algorithm. PSO is a population based search method that imitated the physical movements of the individuals in the swarm as a searching method. Some important PSO characteristics are listed below (Kennedy and Eberhart, 2001; Clerc, 2006):

a. A swarm of particles served as searching agent for a specific solution

b. A particle's position, which consists of some dimensions, is representing a solution of the problem

c. In each iteration step, every particle moves from one position to the next based on its velocity that is equivalent with evaluating different prospective solution of the problem.

d. Particle velocity that is driven the particle movement is affected by the personal best and global best position as well as by random factors.

e. The personal best position of a particle, which expresses the cognitive behavior of the swarm, is defined as the position that gives the best objective function among the positions that have been visited by the particle. Once a particle reaches a position that has a better objective function than the previous best objective function for this particle, the personal best position is updated.

f. The global best position, which expresses the social behavior of the swarm, is the position that gives the best objective function among the positions that have been visited by all particles in the swarm. Once a particle reaches a position that has a better objective function than the previous best objective function for whole swarm, the global best position is also updated.

g. At the end of the iteration process, the global best position is representing the best solution found by the PSO.

The new algorithm for solving the traveling salesman problem is adopting the swarm intelligence philosophy behind the PSO Algorithm. First of all, the proposed algorithm is a population based searching method that is used a swarm of particles as searching agent. Second, a multi dimensional particle is employed in the proposed algorithm to representing any salesman tour as the problem solution. Third, in each iteration step of the proposed algorithm, every particle moves from one position to the next based on specific movement mechanism which is depend on the current position, the personal best position, the global best position, and random factors. Finally, the personal best position of each particles and the global best position of the swarm are always maintained and updated in each iteration step of the proposed algorithm. The building blocks of the proposed swarm-intelligence based algorithm for solving the traveling salesman problem are presented below.

2.1 Representation of Solution

This algorithm employs *n–1* dimensional particle for representing salesman tour of *n* cities. The first city, the starting city of the salesman, is excluded from the particle representation since the salesman tour start and end at this city. Direct representation is employed here, in which each particle dimension can only be filled with the index of the *n–1* available cities and each index can fill exactly one dimension. Some examples of valid solution representation of the traveling salesman problem with $n = 7$ and its salesman tour are presented in Figure 1.

Particle 01: 5-2-3-6-7-4	≡	Tour 01: 1-5-2-3-6-7-4-1
Particle 02: 7-2-5-6-4-3	≡	Tour 01: 1-7-2-5-6-4-3-1
Particle 03: 3-6-4-7-5-2	≡	Tour 01: 1-3-6-4-7-5-2-1
Particle 04: 4-5-3-2-7-6	≡	Tour 01: 1-4-5-3-2-7-6-1
Particle 05: 2-4-5-3-6-7	≡	Tour 01: 1-2-4-5-3-6-7-1

Figure 1. Solution Representation of the TSP with n = 7.

2.2 Basic Particle Movement

In each iteration step, every particle moves from one position to the next based on the current position, the personal best position, the global best position, and random factors. The movement mechanism for the *i*-th particle at the *d*-th dimension can be formally defined in Algorithm 1.

Algorithm 1: Movement Mechanism for the *i*-th Particle at the *d*-th Dimension

1. Select a movement mechanism among current, personal, global, and random mechanism. Random selection is employed here with selection probability of each mechanism are ϕ_c, ϕ_p, ϕ_g, and ϕ_r, respectively for the current, personal, global, and random mechanisms, in which $\phi_c + \phi_p + \phi_g + \phi_r = 1$. If current mechanism is selected go to step 2, if personal mechanism is selected go to step 3, if global mechanism is selected go to step 4, if random mechanism is selected go to step 5.

2. If the current position value of the *i*-th particle at the *d*-th dimension (X_{id}^{t}) has not yet been assigned to other dimension on the next position ($X_{id}^{t} \neq X_{ij}^{t+1}, \forall j, j < d$), assign the next position value of the *i*-th particle at the *d*-th dimension with X_{id}^{t} ($X_{id}^{t+1} = X_{id}^{t}$), otherwise go to step 5.
3. If the personal best position value of the *i*-th particle at the *d*-th dimension (P_{id}) has not yet been assigned to other dimension on the next position ($P_{id} \neq X_{ij}^{t+1}, \forall j, j < d$), assign the next position value of the *i*-th particle at the *d*-th dimension with P_{id} ($X_{id}^{t+1} = P_{id}$), otherwise go to step 5.
4. If the global best position value at the *d*-th dimension (G_d) has not yet been assigned to other dimension on the next position ($G_d \neq X_{ij}^{t+1}, \forall j, j < d$), assign the next position value of the *i*-th particle at the *d*-th dimension with G_d ($X_{id}^{t+1} = G_d$), otherwise go to step 5.
5. Select a random index *u* that has not yet been to other dimension on the next position ($u \neq X_{ij}^{t+1}, \forall j, j < d$) and assign the next position value of the *i*-th particle at the *d*-th dimension with *u* ($X_{id}^{t+1} = u$).

The complete particle movement consists of *r* consecutive execution of Algorithm 1, starting from the first dimension of the particle ($d = 1$) to the last dimension of the particle ($d = n–1$).

3.3. The Algorithm

As mentioned earlier, the proposed algorithm starts by initializing a swarm of multi-dimensional particles followed by a series of iteration steps to move the particles towards the best position. The proposed algorithm is described in Figure 1 and the details of each step and each component will be described afterward.

The proposed swarm-intelligence based algorithm for solving the traveling salesman problem is formally defined as follow:

Step 1. Initialization

In the initialization step, a swarm consisting of *I* particles is initialized by setting the position and personal best value of each particle. Particle position ($X_i^0 = \{X_{i1}^0, X_{i2}^0, \ldots, X_{i,n-1}^0\}$) is being set randomly as a valid tour of *n* cities. Representation of the result of any tour construction heuristics such as nearest neighbor heuristic may be used as initial particle position. The personal best value of each particle is equal to the position value, $P_{id} = X_{id}^0$. Set iteration counter $t = 0$.

Step 2. Iteration

The main part of the proposed algorithm is the iteration step. As shown in Figure 1, each iteration step consists of four sub-steps: decode particle to a salesman tour, evaluate salesman tour performance, update cognitive and social term, and update particle position.

In the first sub-step, each particle position is decoded to a salesman tour. This decoding step is straightforward since direct tour representation is employed in this algorithm.

Then, the performance of each constructed salesman tour is evaluated. This performance value, then, is kept as the fitness value of its corresponding particle. For example, after position of particle i ($\boldsymbol{X}_i^t$) is decoded to the salesman tour Π_i, the performance measurement of salesman tour Π_i, denoted by $Z(\Pi_i)$, is kept as the fitness value of $\boldsymbol{X}_i^t$, denoted by $\varphi\left(\boldsymbol{X}_i^t\right)$, in which $\varphi\left(\boldsymbol{X}_i^t\right) = Z\left(\Pi_i\right)$.

After the fitness value of every particle is determined, the information of the cognitive and social terms of each particle, which are the personal best position and global best position are updated. It is noted that smaller fitness value is desirable, since TSP is a minimization problem. The updating procedure is explained as follows. First, the fitness value of each particle, $\varphi\left(\boldsymbol{X}_i^t\right)$, is compared against its personal best, $\varphi\left(\boldsymbol{P}_i\right)$. The personal best position is set to be the current position, $\boldsymbol{P}_i = \boldsymbol{X}_i^t$, if the fitness value of current position is smaller than its personal best, $\varphi\left(\boldsymbol{X}_i^t\right) < \varphi\left(\boldsymbol{P}_i\right)$. Also, the global best position is set to be the current position, $\boldsymbol{G} = \boldsymbol{X}_i^t$, if the fitness value of current position is smaller than its global best, $\varphi\left(\boldsymbol{X}_i^t\right) < \boldsymbol{G}$. In the last sub-step, the particle position is updated using basic particle movement explained in the section 2.2 (Algorithm 1).

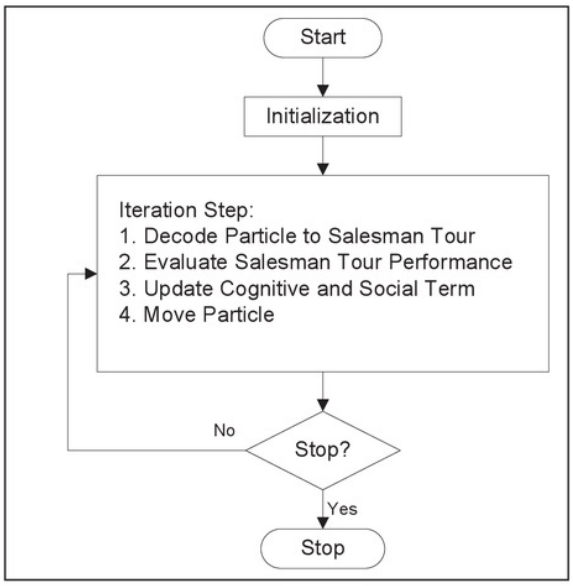

Figure 2: Flowchart of the Proposed Swarm-Intelligence Based Algorithm.

Step 3. Termination

This step will control the mechanism for repetition of this algorithm. The algorithm is terminated whenever the stopping criterion is satisfied; otherwise, the iteration step will be repeated. The

stopping criterion used in this framework is the total number of iteration, i.e. when $t = T$ the algorithm is terminated, otherwise, increase the iteration counter, $t = t + 1$, and repeat Step 2.

Once the algorithm is terminated, the salesman tour represented by the global best position is reported as the best salesman tour found by this algorithm.

3. Computational Experiments

Computational experiments are performed in order to observe behavior of the proposed algorithm and evaluate its performance. The proposed algorithm is implemented in C# language using SharpDevelop 2.0, an open source IDE for .NET Framework, on PC with AMD Athlon II X2 240 2.8 GHz Processor and 1 GB of RAM. Four benchmark data from TSPLIB (Reinelt, 2008) are selected, which are bay29, swiss42, gr48, and brazil58.

First computational experiments are conducted for observing algorithm performance on various parameters value, i.e. number of particles (*I*), number of iterations (*T*), probability of current, personal, global, and random mechanism (ϕ_c, ϕ_p, ϕ_g, and ϕ_r). Three probability settings are tested here: Parameter Setting A with $\phi_c = 0.3$, $\phi_p = 0.3$, $\phi_g = 0.3$, and $\phi_r = 0.1$, Parameter Setting B with $\phi_c = 0.2$, $\phi_p = 0.2$, $\phi_g = 0.2$, and $\phi_r = 0.4$, and Parameter Setting C with $\phi_c = 0.33$, $\phi_p = 0.33$, $\phi_g = 0.33$, and $\phi_r = 0.01$. For each probability setting, several combinations of number of particles and number of iterations are tested. It is noted that the same number of total evaluation is employed for each combination, i.e. 1,000,000 evaluations for each replication runs. For each combination of parameters, each problem is being solved by 10 replications run of the proposed algorithm. The computation results that are recorded are the best tour objective function found (Best Obj.), the average of objective function at the end of each replication (Average Obj.), and the average computational time from 10 replications (Comp. Time). It is noted that the optimal solution for the problem bay29, swiss42, gr48, and brazil58 are 2020, 1273, 5046, and 25395 respectively.

Second computational experiments are conducted for comparing the algorithm performance on random versus structured initialization setting. In the random setting, all particles are set randomly. In the structured setting, some particles are generated using nearest neighbor heuristics (Rego et al., 2011) while the rest are randomly generated. The results are displayed on Table 1, 2, and 3.

4. Discussions and Further Works

The computational results show that the proposed algorithm using specific settings is able to find good solution of the corresponding traveling salesman problem instance. The proposed algorithm on parameter setting C using nearest neighbor initialization is able to obtain optimal solution for instance swiss42, and very close to optimal solution for the other instances.

Table 1. Computational Results on Parameter Setting A

Benchmark Data	No. Particle	No. Iteration	Random Initialization			Nearest Neighbor Initialization		
			Best Obj.	Average Obj.	Comp. Time	Best Obj.	Average Obj.	Comp. Time
bay29	50	20,000	2212	2513.1	6.0	2035	2035.4	6.1
	100	10,000	2301	2578.0	6.0	2026	2034.1	5.8
	200	5,000	2319	2532.6	6.0	2035	2035.0	5.7
	500	2,000	2423	2625.2	6.1	2031	2034.2	5.7
swiss42	50	20,000	1830	2120.9	9.8	1372	1401.0	11.2
	100	10,000	1933	2108.9	9.6	1367	1385.4	10.4
	200	5,000	2051	2196.6	9.8	1380	1387.7	9.8
	500	2,000	1979	2177.2	10.0	1366	1403.1	13.4
gr48	50	20,000	8428	9192.9	12.1	5840	5840.0	15.2
	100	10,000	8266	9084.2	12.3	5840	5840.0	16.5
	200	5,000	8127	9475.7	12.8	5806	5834.8	14.4
	500	2,000	9008	9750.4	12.5	5840	5840.0	16.5
brazil58	50	20,000	42903	47204.5	17.7	27384	27384.0	21.8
	100	10,000	47528	51874.0	17.0	27384	27384.0	23.6
	200	5,000	43865	51133.7	16.4	27384	27384.0	18.9
	500	2,000	46452	52330.3	20.3	27384	27384.0	17.9

Table 2. Computational Results on Parameter Setting B

Benchmark Data	No. Particle	No. Iteration	Random Initialization			Nearest Neighbor Initialization		
			Best Obj.	Average Obj.	Comp. Time	Best Obj.	Average Obj.	Comp. Time
bay29	50	20,000	3247	3499.7	8.8	2134	2134.0	8.7
	100	10,000	3280	3556.2	8.7	2134	2134.0	8.6
	200	5,000	3255	3536.2	8.5	2134	2134.0	8.6
	500	2,000	3263	3563.8	8.5	2134	2134.0	8.6
swiss42	50	20,000	3032	3153.3	15.0	1437	1437.0	18.2
	100	10,000	3054	3187.3	15.1	1437	1437.0	21.0
	200	5,000	3036	3200.9	15.1	1437	1437.0	16.4
	500	2,000	3052	3218.4	15.0	1437	1437.0	15.0
gr48	50	20,000	13290	13873.0	18.5	5840	5840.0	19.5
	100	10,000	13178	13848.7	18.5	5840	5840.0	25.9
	200	5,000	13234	13721.5	19.3	5840	5840.0	23.3
	500	2,000	13534	14001.2	18.9	5840	5840.0	19.5
brazil58	50	20,000	70132	76788.7	29.9	27384	27384.0	29.2
	100	10,000	75691	77535.3	25.8	27384	27384.0	31.0
	200	5,000	70792	76873.3	27.4	27384	27384.0	24.9
	500	2,000	74962	77847.9	37.0	27384	27384.0	24.9

Table 3. Computational Results on Parameter Setting C

Benchmark Data	No. Particle	No. Iteration	Random Initialization			Nearest Neighbor Initialization		
			Best Obj.	Average Obj.	Comp. Time	Best Obj.	Average Obj.	Comp. Time
bay29	50	20,000	2241	2450.9	3.7	2035	2035.0	4.1
	100	10,000	2254	2455.1	3.7	2026	2034.1	3.9
	200	5,000	2350	2482.8	3.8	2035	2035.0	3.7
	500	2,000	2315	2445.9	3.9	2026	2037.2	3.6
swiss42	50	20,000	1695	1834.9	6.0	1273	1281.3	9.1
	100	10,000	1663	1833.2	6.1	1273	1275.9	9.7
	200	5,000	1707	1819.3	6.6	1273	1273.7	10.0
	500	2,000	1574	1829.1	7.2	1273	1273.4	7.0
gr48	50	20,000	6716	7421.7	7.9	5796	5811.0	15.0
	100	10,000	6374	7128.8	9.0	5775	5781.3	9.4
	200	5,000	6687	7269.7	8.8	5775	5785.5	7.8
	500	2,000	6816	7432.6	10.6	5775	5783.4	7.0
brazil58	50	20,000	35771	40534.7	10.8	26480	26763.2	14.7
	100	10,000	37111	40848.9	12.7	26362	26625.3	12.0
	200	5,000	34879	39776.4	16.0	26362	26628.5	10.2
	500	2,000	37900	44073.6	11.8	26362	26693.1	9.0

Three main findings are obtained from the experiments. First, it is better to initialize particle using a tour construction heuristics rather than random initialization. Empirically, the computational result of the proposed algorithm using nearest neighbor initialization are better than the result of the proposed algorithm using random initialization for all instances and all parameter settings. Second, the parameter value of random mechanism selection probability (ϕ_r) has significant effect to the results. Empirically, the smaller value of ϕ_r yields the better results. In addition, in terms of computational efforts the smaller value of ϕ_r also requires the least efforts that are shown by smaller computational time. Third, it is still inconclusive the effect of number of particle and number of iteration on the results. In general, the effect of parameter setting on the computational result needs to be further studied in order to obtain the best parameter setting.

Note that the result on these experiments is gained by pure swarm-intelligence mechanism. Hence, it is possible to improve the result by the addition of effective local search method. Since the local search algorithm is usually exhaustive, it may be done infrequently during the iteration process, for example, perform the local search only to the personal best or global best when the best value is updated by new value. The implementation of local search needs to be further studied.

The computational result over wide range of TSP benchmark data also should be investigated in order to obtain general overview and overall performance of this proposed algorithm.

References

- Bonabeau, E., M. Dorigo, and G. Theraulaz, 1999. *Swarm Intelligence: From Natural to Artificial Systems*. Oxford University Press.
- Chen, S.M. and C.Y. Chien, "Parallelized genetic ant colony systems for solving the traveling salesman problem". *Expert Systems with Applications* 38, 3873-3883.
- Clerc, M. 2006. *Particle Swarm Optimization*. ISTE.

- Dorigo, M. and L.M. Gambardella. 1997. “Ant colonies for the travelling salesman problem”. *BioSystems* 43, 73-81.
- Kennedy, J. and R.C. Eberhart. 2001. *Swarm Intelligence*. Morgan Kaufmann Publishers.
- Laporte, G . 2010. “A concise guide to the Traveling Salesman Problem”. *The Journal of the Operational Research Society* 61, 35-40.
- Rego, C., D. Gamboa, F. Glover and C. Osterman. 2011. “Traveling salesman problem heuristics: Leading methods, implementations and latest advances”. *European Journal of Operational Research* 211, 427-441.
- Reinelt, G.. 2008. *TSPLIB*. http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/
- Tasgetiren, M.F., P.N. Suganthan and Q.K. Pan. 2007. A discrete particle swarm optimization algorithm for the generalized traveling salesman problem. In *Proceedings of GECCO 2007: Genetic and Evolutionary Computation Conference*.
- Zhong, W.L., J. Zhang, and W.N. Chen. 2007. A novel discrete particle swarm optimization to solve traveling salesman problem. In *Proceedings of 2007 IEEE Congress on Evolutionary Computation (CEC 2007)*.

Paper 10

ORIGINALITY REPORT

8%	0%	8%	0%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	"Proceedings of Seventh International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA 2012)", Springer Nature, 2013 Publication	8%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	On		