Computers and Mathematics with Applications 64 (2012) 788–797

Contents lists available at SciVerse ScienceDirect

Computers and Mathematics with Applications

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

Evolutionary algorithm to traveling salesman problems

Yen-Far Liao, Dun-Han Yau, Chieh-Li Chen *

Department of Aeronautics and Astronautics, National Cheng Kung University, Tainan, Taiwan

ARTICLE INFO

Article history:
Received 2 November 2011
Received in revised form 6 December 2011
Accepted 6 December 2011

Keywords:
Particle Swarm Optimization
Traveling Salesman Problem
Fuzzy C-Means clustering
Routing

ABSTRACT

This paper proposed an improved version of the Particle Swarm Optimization (PSO) approach to solve Traveling Salesman Problems (TSP). This evolutionary algorithm includes two phases. The first phase includes Fuzzy C-Means clustering, a rule-based route permutation, a random swap strategy and a cluster merge procedure. This approach firstly generates an initial non-crossing route, such that the TSP can be solved more efficiently by the proposed PSO algorithm. The use of sub-cluster also reduces the complexity and achieves better performance for problems with a large number of cities. The proposed Genetic-based PSO procedure is then applied to solve the TSP with better efficiency in the second phase. The proposed Genetic-based PSO procedure is applied to TSPs with better efficiency. Fixed runtime performance was used to demonstrate the efficiency of the proposed algorithm for the cases with a large number of cities.

1. Introduction

The Traveling Salesman Problem (TSP) was first formulated as a mathematical problem in 1930 and became increasingly popular after 1950. It is one of the most intensively studied problems in optimization even in recent years [1]. The TSP is to find a shortest possible tour that visits each city exactly once for a given list of cities and back to the starting city.

TSP is a well-known NP-hard combinatorial optimization problem. It is used as a benchmark for many optimization methods due to the computational complexity, such as Nearest Neighborhood Search (NNS), Simulated Annealing (SA), Tabu Search (TS), Neural Networks (NN), Ant Colony System (ACS), and Genetic Algorithm (GA) [2]. At present, there are many web sites discussing the Traveling Salesman Problem, and have the benchmark in the standard TSPLIB format [3], such as burma14, berlin52, where the number behind the name represents the number of cities to be studied.

The Particle Swarm Optimization (PSO) is an efficient approach to solve continuous problems. Its applications to lots of optimization problems, such as parameter optimization [4], image processing [5], automatic control [6], production scheduling [7], are presented in different research fields. However, TSPs are a kind of discrete problem and are hard to be solved by conventional PSO. [2] make use of the swap operator to construct the path of TSP.

Pang et al. [8] combined the characteristics of PSO with the concept of Fuzzy theory to find the best path of a TSP. Each particle represents a Fuzzy Matrix, which were used to represent the position and velocity of the particles in PSO and the operators in the original PSO procedure were redefined. Each matrix represents a specific path and is updated by the PSO procedure, until the best solution is obtained.

The aim of this article is to enhance the performance of the PSO by Genetic operators [9] and propose a heuristic approach to solve TSPs in a more efficient manner. The proposed approach can also be applied to job-shop scheduling problems.

* Corresponding author.
E-mail address: chiehli@mail.ncku.edu.tw (C.-L. Chen).

0898-1221/$ – see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.camwa.2011.12.018

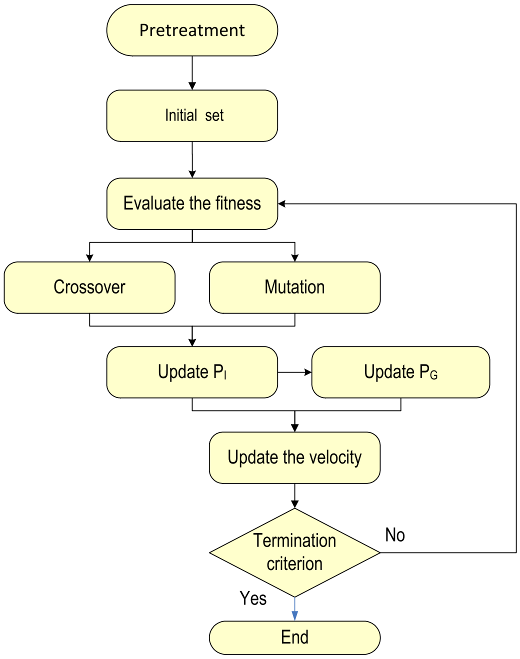


Fig. 1. The procedure of the GPSO.

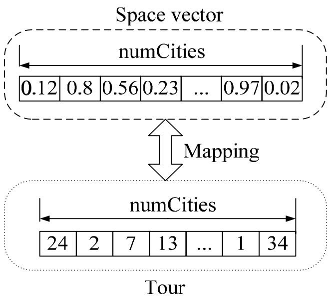


Fig. 2. The mapping of the space vector and the tour.

2. Genetic-based Particle Swarm Optimization (GPSO) algorithm for TSPs

In the PSO algorithm, the swarm is made up of a certain number of particles. Individual memory of a past experience retains the knowledge of where in the search space it performed best. At each iteration, all the particles move in the N-dimensional problem space according to the interaction between individuals and group experience to find the global optimal [10]. The velocity and position of each particle is adjusted by the following formulas:

$$\mathbf{V}_i\,(t+1) = w\mathbf{V}_i\,(t) + c_1 r_1\,(P_i - \mathbf{X}_i(t)) + c_2 r_2\,(P_G - \mathbf{X}_i(t))\,, \tag{1}$$

$$\mathbf{X}_i\,(t+1) = \mathbf{X}_i(t) + \mathbf{V}_i\,(t+1)\,, \tag{2}$$

where i denotes the i-th particle in the swarm, t is the iteration number, $\mathbf{V}_i$ is the velocity vector of the i-th particle, $\mathbf{X}_i$ is the position vector of the i-th particle, P_i is the local best position that particle i had reached, and P_G is the global best position that all the particles had reached. r_1 and r_2 are random numbers between 0 and 1. w is called the inertia weight. c_1 and c_2 are two constant numbers, which are often called the cognitive confidence coefficients. In order to improve the information change between particles, a Genetic-based PSO is proposed as shown in Fig. 1.

Due to the discrete nature of the TSP, each particle represented by a set of floating decimal space vectors denotes a tour of n cities through a mapping function [11], where a higher decimal value leads to a higher priority of visiting order, as shown in Fig. 2. The crossover mechanism in the GPSO algorithm is carried out by a random process, which selects two random particles for producing two offspring, as shown in Fig. 3. Mutation is a process in which vector value may be altered randomly. In the GPSO algorithm, a randomly chosen particle is perturbed by altering the value of its corresponding space vector to generate an offspring as demonstrated in Fig. 4.

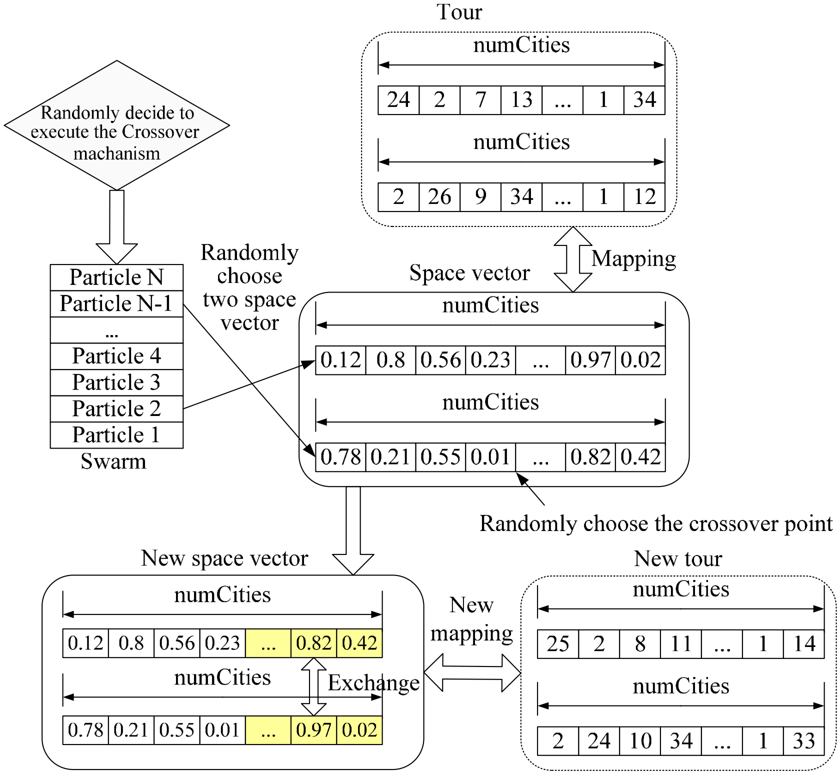


Fig. 3. Schematic diagram of the crossover process in the GPSO approach.

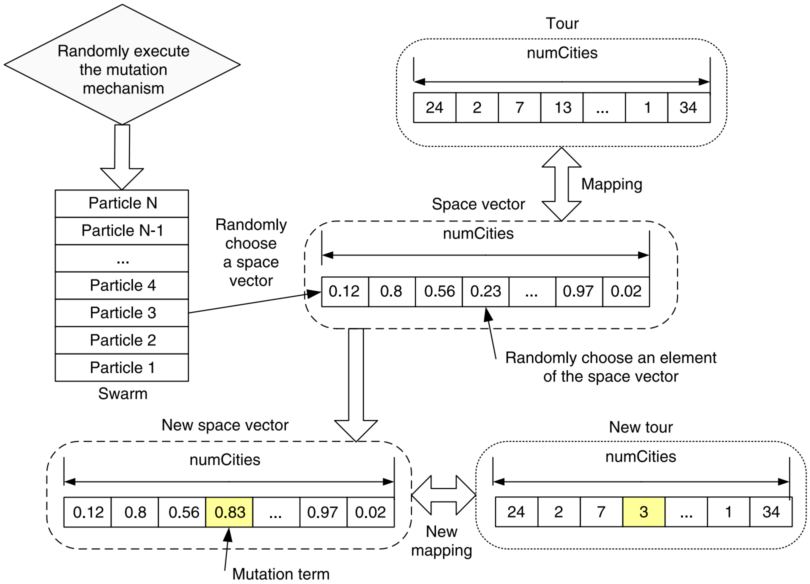
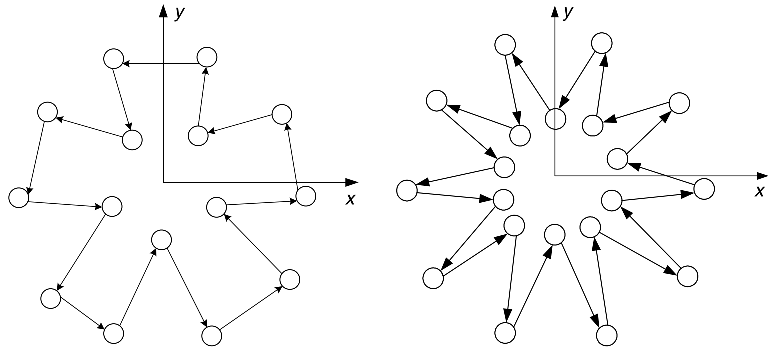


Fig. 4. Schematic diagram of a mutation process in the GPSO approach.

3. Rule-based tour path generation for TSPs

3.1. City clustering and initial tour path generation

A direct use of a random search method, such as the Genetic algorithm and PSO, for TSPs of a large city cluster will lead to computational inefficiency and may not even result in a suboptimal solution. For a very large cluster, cities considered can be divided into sub-clusters before using the PSO algorithm. The fuzzy C-Means clustering method proposed by Bezdek [12]



(a) Non-crossing path. (b) Stellate path.

Fig. 5. Tour path generated by step 3.

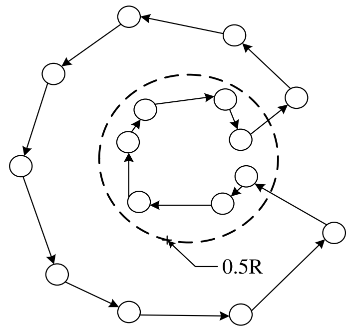


Fig. 6. C-ring path obtained by inner and outer clusters.

is an enhanced approach for the clustering procedure. The use of the Fuzzy C-Means clustering method will reduce the complexity of the searching procedure by integrating sub-clusters. For example, an initial solution for a 500-city problem can be generated through a merge process using initial solutions of five 100-city sub-clusters.

For the single cluster case, a procedure to generate an initial tour path without crossover is proposed as follows.

Step 1: Calculate the geographical center (C_x, C_y) of the city cluster.

$$C_x = \frac{\sum_{i=0}^{N} p_{ix}}{N}, \tag{3}$$

$$C_y = \frac{\sum_{i=0}^{N} p_{iy}}{N}, \tag{4}$$

where (p_{ix}, p_{iy}) represents the coordinate of the i-th city, and N represents the number of cities.

Step 2: Calculate the orientation angle of city to the geographical center (C_x, C_y).

$$\mathrm{ang}_i = \frac{180}{\pi} \tan^{-1}\left(\frac{p_{iy} - C_y}{p_{ix} - C_x}\right), \quad \text{where } 0 \le \mathrm{ang}_i \le 360. \tag{5}$$

Step 3: Generate a tour path according to the orientation angle value.

In general, this process results a non-crossing path as demonstrated in Fig. 5. A stellate path may be obtained for some geographical distribution, as shown in Fig. 5(b), if only orientation angle is considered. In this case, step 3.1 can be applied instead.

Step 3.1: Generate a C-ring path by orientation angle value and the distance from the geographical center. In this step, cities are divided into to two clusters, i.e. the outer cluster and the inner cluster, according to their distance to the geographical center(C_x, C_y). Step 3 is applied for both clusters and then two tour paths are connected to form a C-ring path, as shown in Fig. 6, where R represents the average distance between cities and the geographical center(C_x, C_y).

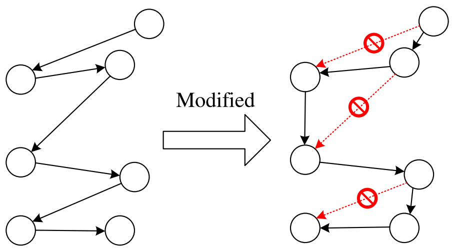


Fig. 7. Shorter path obtained by removing jagged path.

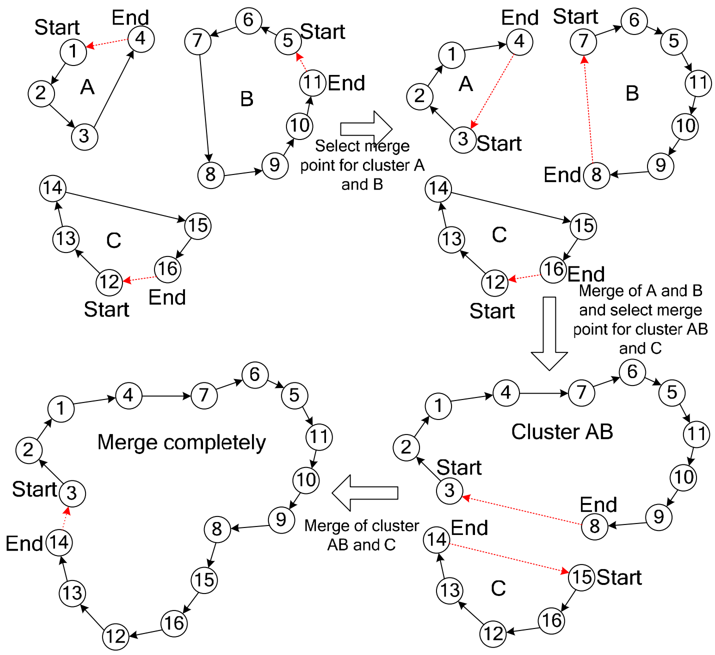


Fig. 8. The merge process of sub-clusters.

Step 4: Reduce the jagged path. Although a non-crossover tour path can be generated by step 3, there is no guarantee that it will be the shortest path. A swap strategy, which randomly exchanges the path between the nearby nodes, is applied subsequently to generate a shorter path as shown in Fig. 7.

3.2. Merge of city clusters

For multi-cluster cases, initial tour paths for sub-clusters can be generated using the procedure proposed in Section 3.1. To integrate the tour paths of sub-clusters without crossover will be important to the efficiency of solving TSPs. A procedure to merge tour paths of sub-clusters is as follows.

Step 1: Select the largest sub-cluster as the main cluster, and then merge its tour path with the nearest sub-cluster.
Step 2: Select cities for the merge operation according to the distance between cities of two sub-clusters. Take Fig. 8 as an example, sub-cluster B is the main cluster and city A4 and city B7 are the nearest cities between sub-cluster A and B. Then A4 and B7 are selected to be A-End and B-Start, respectively, where A-End represents the last city of the tour path of sub-cluster A and B-Start represents the first city of the tour path of sub-cluster B.
Step 3: Choose A-Start and B-End among the nearby cities of A-End and B-Start with shortest distance. In. Fig. 8, A1 and A3 are the candidates of A-Start; B6 and B8 are the candidates of B-End. Since distance(A3, B8) is the shortest among distance(A1, B8), distance(A1, B6) and distance(A3, B6), A3 and B8 are then selected as the A-Start and B-End, respectively.
Step 4: Merge both tour paths by connecting A-Start with B-End and A-End with B-Start.
Step 5: Repeat the merge operation until all sub-clusters are merged together.

This procedure will avoid crossover between sub-clusters. Based on the GPSO method proposed in Section 2, the TSP can be solved by the procedure shown in Fig. 9.

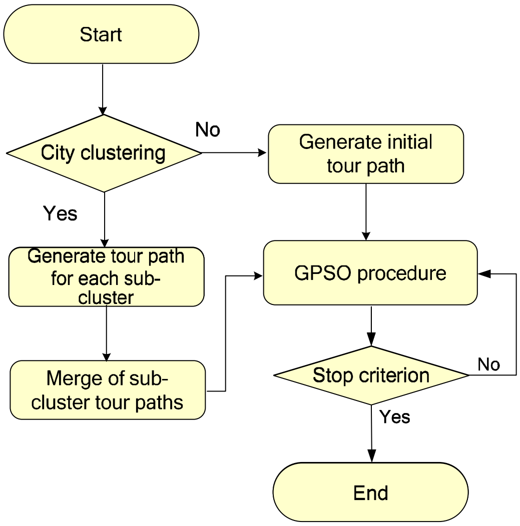


Fig. 9. Algorithm for solving TSPs.

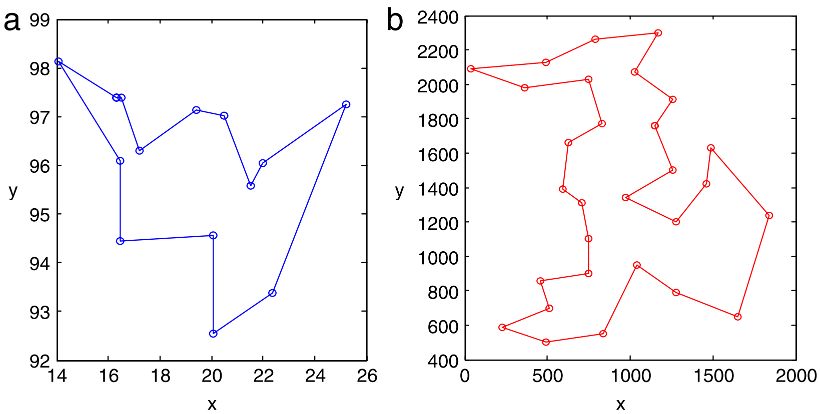


Fig. 10. (a) Optimal solution for burma14. (b) Optimal solution for bayg29.

Table 1
Results for burma14 by conventional PSO with space vector representation.

Test case	1	2	3	4	5	6
Particle	100	100	100	100	300	300
Iteration	100	100	100	100	100	300
w	0.5	0.5	0.5	0.75	0.75	0.75
c_1	0	0	0.5	0.5	0.5	0.5
c_2	3	2	2	2	2	2
Ave. tour distance	31.8289	31.4833	31.4501	31.5092	31.3083	31.1862
Ave. computation time (ms)	275.78	270.62	259.06	258.28	797.18	2521.71
Optimum results/100 trials	35	48	53	54	67	72

4. Cases study for TSPs

In this section, all numerical studies were carried out on a PC with an Intel® Core2 Duo CPU T7300 2.0 GHz.

Case 1: Single cluster by PSO with vector space transformation.

This case considers a 14-city case (burma14) and a 29-city case (bayg29) using PSO approach with vector space transformation. Fig. 10(a) and (b) show the optimal solutions of burma14 and bayg29, respectively. Tables 1 and 2 shows the computation parameters and their corresponding results obtained from 100 trials. It illustrates that optimal solution

Table 2
Results for bayg29 by conventional PSO with space vector representation.

Test case	1	2	3	4	5	6
Particle	100	100	100	100	300	300
Iteration	100	100	100	100	100	300
w	0.2	0.5	0.25	0.25	0.5	0.25
c_1	0.2	0.2	0.3	0.3	0.2	0.2
c_2	2	2	2	2	2	2
Ave. tour distance	11719.6	12014.8	11593.8	13925.4	11131.2	10256.1
Ave. computation time (ms)	448.28	438.28	440	447.81	1330.31	4044.53
Optimum results/100 trials	0	0	0	0	0	1

Table 3
Results for berlin52 by the proposed GPSO.

Test case	1	2	3	4	5	6
Particle	100	100	100	100	100	100
No. of iteration	100	100	100	100	100	80
c_2	3	3	3	2	2	2.5
w	0.25	0.25	0.25	0	0.25	0.25
Prob. crossover	0.2	0.8	0.8	0.2	0.2	0.2
Prob. mutation	0.5	0.5	0.2	0.5	0.5	0.75
Mutation gain	2	0.5	0.5	5	5	5
Ave. tour distance	7553.86	7555.79	7549.67	7546.64	7545.12	7545.12
Ave. computation time (ms)	814.68	865	812.04	817.64	811.88	700.77
Optimum results/100 trials	91	92	93	97	99	99

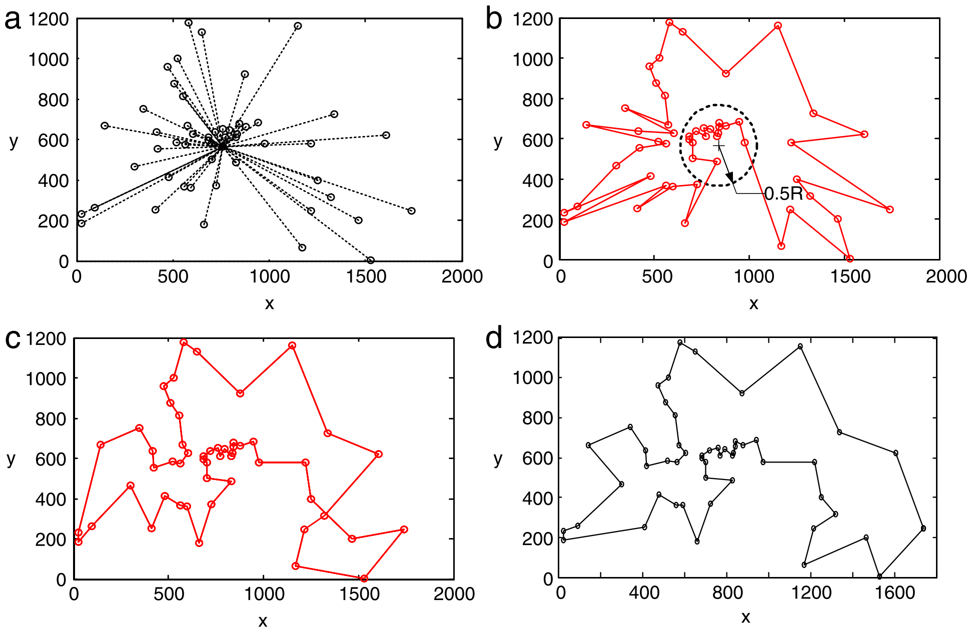


Fig. 11. (a) Calculating the geographical center and the distance between the cities. (b) C-ring path obtained by step 3. (c) Jagged free path by step 4. (d) Optimal tour path obtained by GPSO (7544.3659).

achieving rate for burma14 is up to 70% within 100 trials. However, to reach an optimal solution for bayg29 is difficult to achieve with 300 iterations using the PSO approach with vector space transformation.

Case 2: Single-cluster case (berlin52).

Fig. 11 demonstrates the generation of tour path for a 52-city case by the proposed procedure. Due to the introduction of the mutation process, better performance can be obtained with a dominant weighting of c_2, i.e. $c_1 = 0$. Table 3 shows the parameters of the proposed GPSO algorithm and their corresponding results obtained from 100 trials. The optimal solution obtained with tour distance 7544.3659 is shown in Fig. 11(d). Compared with the approach applied in Case 1, the proposed approach has a better performance on accuracy and computation efficiency.

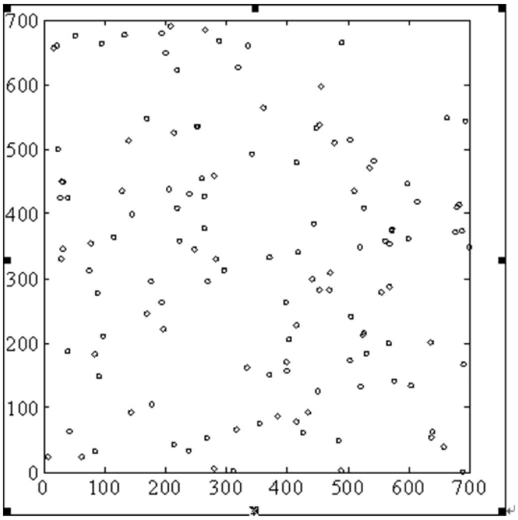


Fig. 12. The distribution of 130 cities.

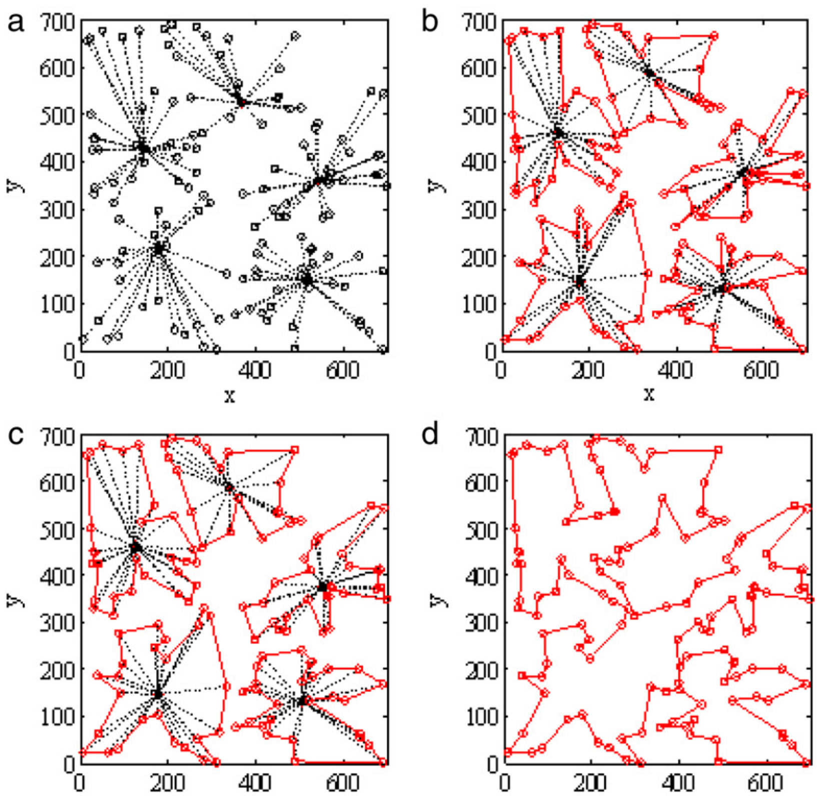


Fig. 13. Generation of tour path for 130-city case (a) Clustering. (b) Initial tour path of each sub-cluster. (c) Jagged free path of each sub-cluster. (d) Merged tour path (distance 6967.3338).

Case 3: Multi-cluster case (130 cities).

Fig. 12 shows the distribution of 130 cities for the TSP. Cluster the cities into 5 sub-clusters, as shown in Fig. 13(a). The initial tour path obtained by the merge operation is shown in Fig. 13(b)–(d). Fig. 14 shows the improved result obtained by the proposed approach.

Case 4: Multi-cluster case (1002 cities).

This case considers the influence of sub-cluster number using a 1002-city case with the same GPSO parameters. Fig. 15 shows the average best solution obtained by 10 trials for different sub-cluster number. It reveals that the increasing of

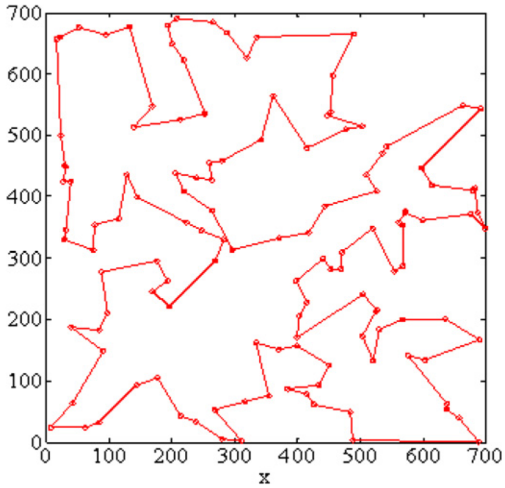


Fig. 14. Tour path obtained by the proposed approach (distance measure 6570.4912).

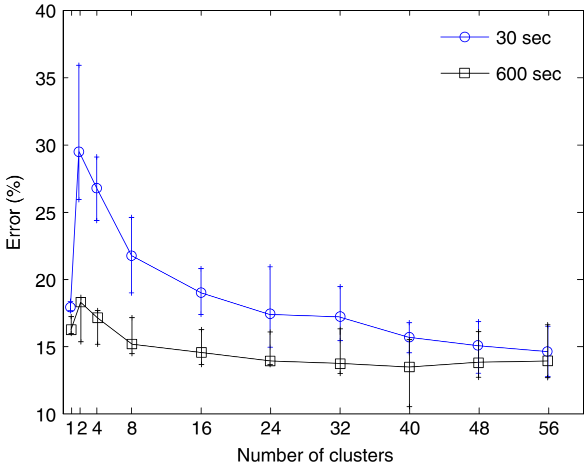


Fig. 15. Average performance obtained for 1002-city case.

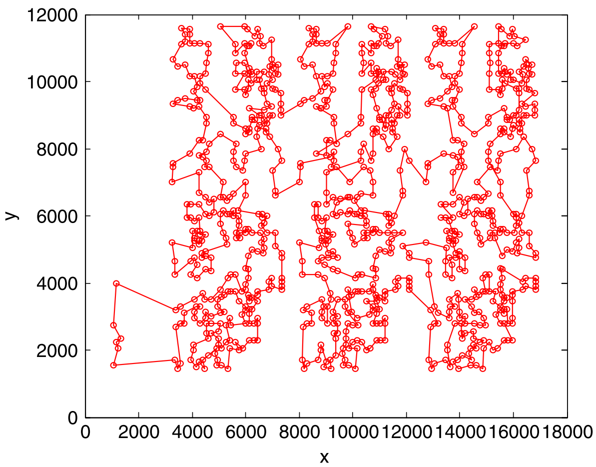


Fig. 16. Best tour path obtained for 1002-city case with 40 sub-clusters in 600 s. (Total distance of tour path is 286139, which is 10.45% larger than the optimum.)

computation effort would not lead to significant improvement of the resulting solution for single cluster approach. The use of a single cluster also leads to less variance in 10 trials. In general, the computation effort for generating an initial tour path is very small and can be neglected. 600 s is sufficient for achieving reasonable solution for problem with different sub-cluster number. Appropriate increase of sub-cluster will obtain a better solution with less computation power. In this case, dividing whole cities into 40–56 sub-clusters results in a reasonably good solution within 30 s, which is difficult to achieve using the existing PSO approaches. The best solution obtained using 40-subcluster in 600 s is shown in Fig. 16.

5. Conclusions

In this paper, the GPSO algorithm was proposed to solve TSPs. A rule-based algorithm was introduced to generate an initial tour path, which sorts the visit sequence based on the azimuth angle to the geographical center. Fuzzy C-Means was also applied to reduce the complexity of solving TSPs with large number of cities, and the solution is obtained by merging of sub-cluster tour paths and the proposed GPSO algorithm. The results show that the GPSO with sub-cluster integration leads to a better efficiency in dealing the case with large number of cities. Although there is no guarantee that an optimal solution can be obtained, a reasonably good solution can be obtained with proper cluster number within a computation time limitation, as illustrated in the study cases. The proposed approach is beneficial to path planning of goods delivery.

Acknowledgment

The work was partly supported by the National Science Council, Taiwan, under grant No. NSC97-2221-E006-181.

References

[1] R. Bello, Y. Gomez, A. Nowe, M.M. Garcia, Two-step particle swarm optimization to solve the feature selection problem, in: Proceedings of International Conference on Intelligent Systems Design and Applications, 2007, pp. 691–696.
[2] K.P. Wang, L. Huang, C.G. Zhou, W. Pang, Particle swarm optimization for traveling salesman problem, in: Proceedings of International Conference on Machine Learning and Cybernetics, vol. 3, 2003, pp. 1583–1585.
[3] G. Reinelt, TSPLIB, Institute of information, University Heidelberg, 2008. http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/.
[4] J.F. Chen, Z.W. Ren, X.N. Fan, Particle swarm optimization with adaptive mutation and its application research in tuning of PID parameters, in: Proceedings of 1st International Symposium on Systems and Control in Aerospace and Astronautics, 2006, pp. 1–5.
[5] K. Chandramouli, E. Izquierdo, Image classification using chaotic particle swarm optimization, in: IEEE International Conference on Image Processing, 2006, pp. 3001–3004.
[6] A. Chatterjee, K. Pulasinghe, K. Watanabe, K. Izumi, A particle swarm optimized fuzzy-neural network for voice-controlled robot systems, IEEE Transactions on Industrial Electronics 52 (3) (2005) 1478–1489.
[7] S. Chandrasekaran, S.G. Ponnambalam, R.K. Suresh, N. Vijayakumar, A hybrid discrete particle swarm optimization algorithm to solve flow shop scheduling problems, in: Proceedings of IEEE Conference on Cybernetics and Intelligent Systems, 2006, pp. 1–6.
[8] W. Pang, K.P. Wang, C.G. Zhou, L.J. Dong, Fuzzy discrete particle swarm optimization for solving traveling salesman problem, in: Proceedings of the fourth International Conference on Computer and Information Technology, 2004, pp. 796–800.
[9] M.A. Shoorehdeli, M. Teshnehlab, H.A. Moghaddam, Feature subset selection for face detection using genetic algorithms and particle swarm optimization, in: Proceedings of International Conference on Networking, Sensing and Control, 2006, pp. 686–690.
[10] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proceedings of IEEE International Conference on Neural Networks, vol. 4, 1995, pp. 1942–1948.
[11] W. Pang, K.P. Wang, C.G. Zhou, L.J. Dong, M. Liu, H.Y. Zhang, J.Y. Wang, Modified particle swarm optimization based on space transformation for solving traveling salesman problem, in: Proceedings of International Conference on Machine Learning and Cybernetics, vol. 4, 2004, pp. 2342–2346.
[12] J.C. Bezdek, Fuzzy mathematics in pattern classification, Ph.D. Thesis, Cornell University, 1973.