An Algorithm of Set-Based Differential Evolution for Traveling Salesman Problem

Tao Liu[1]
Michiharu Maeda[2]
Fukuoka Institute of Technology, 3-30-1 Wajiro-higashi, Higashi-ku, Fukuoka 811-0295, Japan

***Abstract*—This paper is concerned with combinatorial optimization problems (COPs) for traveling salesman problem (TSP). Differential evolution (DE) is a population-based stochastic technique of evolutionary algorithm (EA), which has been widely used to solve COPs over continuous space in many scientific and engineering fields. Set-based differential evolution (SBDE) is based on a set-based representation scheme that enables differential evolution (DE) to characterize the discrete search space. A parameter ω indicating the possibility is added into the formula for the mutation. All arithmetic operators for elements, individuals, vectors and are replaced by new definitions. In this paper, SBDE is amplified in detail for TSP of COPs in discrete space. The possibility ω is tested and the range allowed of ω is identified. The performance of SBDE with the range allowed of ω is evaluated on TSP. The results of the numerical experiments show that the convergence of SBDE is fast and SBDE is effective in quality for TSP.**

Keywords— differential evolution, traveling salesman problem, discrete combinatorial optimization problem.

I INTRODUCTION

Evolutionary algorithms (EAs) are a field of evolutionary computation (EC) and have been successfully applied in search and optimization techniques in diverse fields. EAs are inspired by the natural evolution of species and solve complex problems by mimicking the processes of Darwinian evolution, such as reproduction, mutation, recombination, and selection. Evolution can be seen as a process leading to the maintenance or increase of a population's ability to survive and reproduce in a specific environment. This ability is called evolutionary fitness. Evolutionary fitness can be viewed as a measure of the organism's ability to anticipate changes in its environment. The EA is started with an initial population of individuals generated randomly. Each individual is then assigned a fitness calculated using the fitness function. Individuals of the population with high fitness scores represent better solutions to the problem than the ones with lower fitness scores. Then, until termination conditions are satisfied, reproduction, mutation, recombination, and selection are repeated. The techniques of EAs are similar, but the implementation details of EAs are different. There are many types of EAs. The most popular type of EA is genetic algorithm (GA). There is also differential evolution (DE) [1] [2].

Optimization problems have received much attention in diverse fields, such as economics, engineering, marketing and so on. EAs have been applied to do the global best solution search. Optimization problems can be classified into two main classes, which is continuous combinatorial optimization problems (COPs) and discrete COPs. Differential evolution (DE) is, proposed by Storn and Price, is one of the novel EAs and has been successfully applied in diverse fields [3]. DE is a simple population-based stochastic technique, which is efficient and effective in the continuous domain. Predominately it is used to fine solutions in a continuous space. As much optimization problems are defined in the discrete space, research on the DE solution search in a discrete space has appeared.

The algorithm that set-based differential evolution (SBDE) is based on a set-based representation scheme that enables DE to characterize the discrete search space. All arithmetic operators for elements, individuals, vectors and are replaced by new definitions. The evolution strategies of mutation, crossover and selection make the candidate solutions diversities. [4]

Traveling salesman problem (TSP) is a well-known and typical example of COPs. In TSP, there is a salesman and cities are given. The goal is to find out the shortest Hamilton cycle that the salesman visits all the cities once and in the end returns to the starting city. Actually, TSP is known as a NP-hard problem in COPs and studied in operations research and theoretical computer science. [5]

In this paper, SBDE is amplified in detail and applied to the new benchmark comparisons of TSP. A parameter ω indicating the possibility is added into the formula for the mutation. The possibility ω is tested and the range allowed of ω is identified. The results of the numerical experiments show that SBDE with the range allowed of ω is effective in quality for TSP.

II DIFFERENTIAL EVOLUTION

Differential evolution (DE) is a population-based stochastic algorithm. It is similar to genetic algorithm (GA) in theory and includes three main operations: crossover, mutation and selection. It is used as an optimizer in continuous space. In DE, a population of N_P D-dimensional $i \in [1,2,..,N_P]$ vector $\mathbf{x} = (x_1, x_2, ..., x_n)$, also called individuals or target vector, are initialized within the initial parameter bounds randomly and the objective function value is evaluated. Then, the optimization process is carried out [6] [7].

a) Mutation

In mutation, DE generates new parameter vectors by adding the weighted difference between two randomly chosen vectors

978-1-4799-5955-6/14/$31.00 ©2014 IEEE

to a third vector. One of the most frequently used mutation strategies is "DE/rand/1", which is used in this paper. By mutation, a new vector $\boldsymbol{v}$ is generated as follow:

$$\boldsymbol{v} = \boldsymbol{x}_a + r_{rand} \times (\boldsymbol{x}_b - \boldsymbol{x}_c) \quad (1)$$

where $a,b,c \in [1,2,..,N_P]$ and $a \neq b \neq c \neq i$ are generated randomly. $r_{rand} \in [0, 2]$ is a real and constant factor for scaling the difference vector.

b) Crossover

Crossover is employed to increase the diversity in the searching process. The trial vector $\mathbf{x}^{new} = (x_1^{new}, x_2^{new}, ..., x_n^{new})$ is generated as follow:

$$\boldsymbol{x}^{new} = \begin{cases} \boldsymbol{v} \ (U \leq R) or \ (j = k) \\ \boldsymbol{x} \qquad otherwise \end{cases} \quad (2)$$

where the crossover factor $R \in [0, 1]$ controls the diversity of the population. j is the parameter index, which ensure that the trial vector $\mathbf{x}^{new}$ will differ from its corresponding target vector $\mathbf{x}$ by at least one parameter. $k \in [1,2,\ldots,N_P]$ and $U \in [0,1]$ are generated randomly.

c) Selection

The selection is employed to compare the fitness function value of the target vector and the trial vector to determine who can survive in the next generation.

$$\boldsymbol{x} \leftarrow \boldsymbol{x}^{new}, if\big(f(\boldsymbol{x}^{new}) \geq f(x)\big) \quad (3)$$

where f represents the fitness function for maximization optimization. [8]

III *Set-based Differential Evolution*

a) Redefinition

The classical DE is an effective approach to global optimization over continuous space. In order to extend the application of DE to COPs with discrete space, SBDE is proposed in this paper. In SBDE, element, individual, vector and all related arithmetic operators are all refined as follow [9][10]:

1. Element: an element *(x, y)* represent an edge *(x, y)* connected city x and city y. For example, element (1, 2) represents an edge from the city 1 to the city *2*.
2. Individual: a individual x_1 represents a feasible solution. For example, if there are 3 cities, the individual x_1 = {(3, 2), (2, 1), (1, 3) represents a Hamiltonian circuit. The circuit is $3 \rightarrow 2 \rightarrow 1 \rightarrow 3$.
3. Vector $\mathbf{x}=(x_1 \cup x_2 \cup \ldots \cup x_n)$ represents subsets of candidate solutions.
4. Multiplication Operator: *rand* $\in [0, 1] \times$ element: *rand* $\times$ *(x, y) = rand(x, y)*. *rand(x, y)* represents the element *(x, y)* is used to constructed a feasible solution with the possibility of rand. For example, $0.3 \times (2, 1) = 0.3(2, 1)$. In TSP, 0.3(2, 1) represents the edge (2, 1) is visited with the possibility of 0.3.
5. Subtraction Operator: individual x_1 - individual x_2 = element $\in$ individual x_1, but element $\notin$ individual x_2. For example, if in TSP, x_1= {(1, 2),(2, 3),(3, 4),(4, 5),(5, 1)}, x_2 ={(1, 2),(2, 4),(4, 3),(3, 5),(5, 1)}, $x_1 - x_2$= {(2, 3),(4, 5)}.
6. Addition Operator: elements + elements: If the elements in different individuals are the same, the possibilities are set to the largest one. If the elements in different individuals are not the same, all the elements are put into one individual. For example, if in TSP, {0.2(1, 2), 0.13(3, 4)} + {0.9(1, 2), 0.99(4, 2)} = {0.9(1, 2), 0.13(3, 4), 0.99(4, 2)}.

In order to characterize the discrete search space of COPs, Eq.(1) is deformed as follow in SBDE.

$$\boldsymbol{v} = \omega \times \boldsymbol{x}_a + r_{rand} \times (\boldsymbol{x}_b - \boldsymbol{x}_c) \quad (4)$$

where ω indicates the possibility and is generated randomly.

b) Individual Updating

At the beginning of the algorithm, vector $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ is initialized with feasible candidate solutions generated randomly. Then, the fitness of individuals is evaluated. According to the mutation, a new vector is obtained. Each individual of the trail vector is a candidate solution. In mutation, a possibility for each element in each individual is generated randomly. According to Eq. (4), ω and $r_{rand} \in (0, 1)$ are the possibilities and generated for each element. In each generation, a random number α is generated for each individual. For each element, the one whose possibility is not smaller than α is reserved to a set. Here, this set is called remain set. For example, in TSP, the elements {0.8(1, 2), 0.32(4, 5)} are given. If α = 0.5, the element {(1, 2)} whose possibility is larger than α = 0.5 is reserved to the remain set. To construct a feasible solution, a Hamiltonian circuit, in the process of mutation, elements are learned from the remain set for each individual. If the construction of a feasible solution is not finished and there is no available element in remain set, heuristic-based selection is applied to finish the feasible solution construction. In this paper, nearest neighbor search is applied [11][12]. After the mutation, according to the crossover operator, a trail vector is generated. Each individual of the trail vector is evaluated. In the process of selection, if the trail vector yields an adaptive fitness for the problem than the target vector individual, the newly trail vector will be accepted. Until the termination conditions are satisfied, the mutation, crossover and selection operators are implemented. The scheme of SBDE is shown in Fig.1.

Procedure of SBDE algorithm:
Set parameters, initialize population
  Evaluation
while (termination condition not met) do
  Mutation
  α generated randomly
  Remain set constructed
  Circuits constructed
  Crossover
  Selection
  Evaluation
end while
exit

Fig.1 Scheme of SBDE

978-1-4799-5955-6/14/$31.00 ©2014 IEEE

Table 1 Results for different values of ω

ω	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1	ω_{rand}
Mean	448.932	454.328	458.559	459.480	461.280	463.953	463.406	471.534	477.810	494.091	1398.777	430.362
Best	435.663	442.898	441.237	448.136	446.156	448.338	447.039	456.163	452.194	476.806	507.996	428.982
Worst	464.822	464.074	465.740	472.917	472.573	485.778	474.310	490.108	505.774	505.774	1546.480	435.781

IV NUMERICAL EXPERIMENT

The accuracy is compared among genetic algorithm (GA), ant colony optimization (ACO), max-min ant system (MMAS), set-based particle swarm optimization (SPSO), and SBDE for the instances of eil51, st70, eil76, kroA100, and eil101 from the traveling salesman problem library (TSPLIB) [13].

a) Parameters Influence

The population size of SBDE can also influence the performance of the algorithm. In general, a small size is able to accelerate the search process, but the diversity of the population is reduced. On the other hand, a large population size brings higher diversity to the population, but the computational effort in each iteration significantly increases, and thus the search process becomes slow. For TSP, it can be seen that size=20, 30, and 40 are able to achieve the best results according to traditional studies. In the experiments this paper, the intermediate value M=30 is applied.

For ω, if ω is small, there will be fewer elements in the remain set to construct a feasible solution. Oppositely, if ω =1, the elements in the previous **V** will not be forgotten, and thus there will be too many elements in the remain set for the elements to learn from. To see the influence of ω in SBDE, the cases of eil51 is applied as the testing problem. All the simulations in the comparison are deployed the same parameters setting except ω. The maximum number of iterations allowed for the testing problem is 1500. For each selected ω, the results in Table1 are the averages of 30 runs. $\omega_{rand} \in (0, 1)$ is generated randomly for each element in the individuals.

From the Table 1, we can see that when $\omega_{rand} \in (0, 1)$ is generated randomly for each element in the individual, the values of mean, best and worst gained are smallest. In the numerical experiments, $\omega_{rand} \in (0, 1)$ is adopted. [14]

b) Superiority of Set-based Techniques

ACO and MMAS both start from a complete initial solution and try to find a better solution from a neighborhood of current solution. The ants get new solutions in local changes. In fact, the ants update their routes with the pheromone. Therefore, ACO and MMAS lapse into local optimal solutions.

Set-based techniques, both SPSO and SBDE, construct solutions for some elements of the solutions which have been gotten. To construct a feasible solution, a Hamiltonian circuit, a selection operator is applied. In this paper, nearest neighbor search is applied. Therefore, the new solutions are composed by some elements of the solutions which have been gotten and some elements which are gotten by selection operator. The diversity of solutions is kept. It is not to lapse into local optimal solutions easily for set-based techniques.

c) Computation Time

The worst case for ACO, MMAS, SPSO and SBDE of the computational complexity of generating a solution is $O(n^2)$. SBDE is improved by SPSO, so the route updating method of SPSO and SBDE is the same. The remain set is used in both SPSO and SBDE. In both SPSO and SBDE, the elements are selected from the remain set, not all the candidate cities. The largest possibility element in the remain set is selected first. The selection operator in both SPSO and SBDE always applies a much smaller set with the search process going on. Therefore, the execution time is shortened. But for ACO and MMAS, the city is selected from all the unselected cities whose pheromone is largest for the ant. The ants get easy to trap into local optima. With the search process going on, the pheromone updating is computed, but the possibilities used in SPSO and SBDE are generated randomly and do not need updaing. Therefore, the execution time of SPSO and SBDE is shorter than ACO and MMAS.

Although the route updating method is the same for SPSO and SBDE, the updating formulas are different. The update formula of SPSO is shown below:

$$v_{t+1} = \omega \times v + c_1 \times r_1 \times (p_{id} - x_t) + c_2 \times r_2 \times (p_{gd} - x_t) \quad (5)$$

where ω is the inertia weight factor, r_1 and r_2 are random value in the range [0, 1], v and x_t are the current velocity and position of the particle, respectively, p_{id} is the best solution this particle has reached, p_{gd} is the current global best solution of all the particles, c_1 and c_2 are learning factors.

The remain set of SPSO is constructed by elements of v, $p_{id} - x_t$ and $p_{gd} - x_t$. According to Eq.(4), The remain set of SBDE is constructed by elements of $\boldsymbol{x}_a$, $\boldsymbol{x}_b - \boldsymbol{x}_c$. The remain set of SBDE is smaller than the remain set of SPSO. Therefore, the execution time of SBDE is shortest among ACO, MMAS and SPSO.

d) Performance on the TSP

Table 2 and Fig.2 show, for eil51, the results of the comparison among GA, ACO, MMAS, SPSO, and SBDE.

978-1-4799-5955-6/14/$31.00 ©2014 IEEE

Table 2 Results for eil51

obtained by GA, ACO, MMAS, SPSO and SBDE

Algorithm	Worst	Best	Mean
GA	1643.82	1241.60	1452.71
ACO	464.40	438.74	448.52
MMAS	450.68	438.74	444.86
SPSO	456.82	428.98	439.34
SBDE	441.87	428.87	430.19

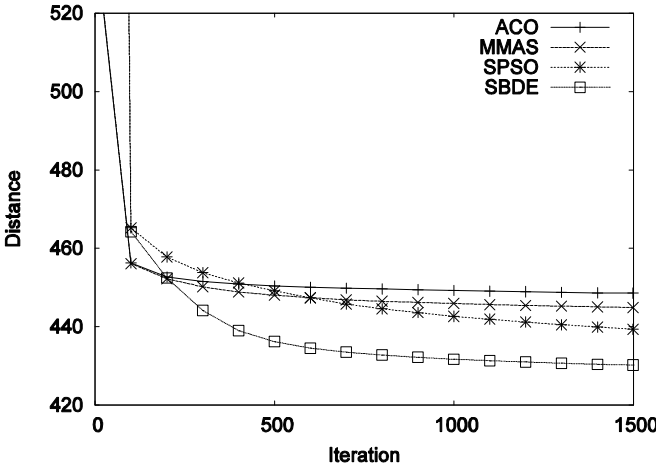

Fig.2 Results of mean value for eil51

obtained by ACO, MMAS, SPSO and SBDE

The optimal value in the case of eil51 is 426. The results in Table 2 and Fig.2 are the averaged 1000 runs. The maximum number of iterations in each run is 1500. It can be seen clear that the worst value, the best value and the mean value gained by the proposed algorithm SBDE are the minimum among the methods in Table 2 and the convergence of SBDE is fastest than the others in Fig.2. SBDE is the most effective in quality than the others here and the convergence of SBDE is the fastest.

Table 3 and Fig.3 show, for st70, the results of the comparison among GA, ACO, MMAS, SPSO, and SBDE.

Table 3 Results for st70

obtained by GA, ACO, MMAS, SPSO and SBDE

Algorithm	Worst	Best	Mean
GA	3542.34	2818.95	3222.28
ACO	720.39	692.22	707.88
MMAS	717.24	692.22	707.66
SPSO	719.04	677.80	695.24
SBDE	693.93	677.11	679.44

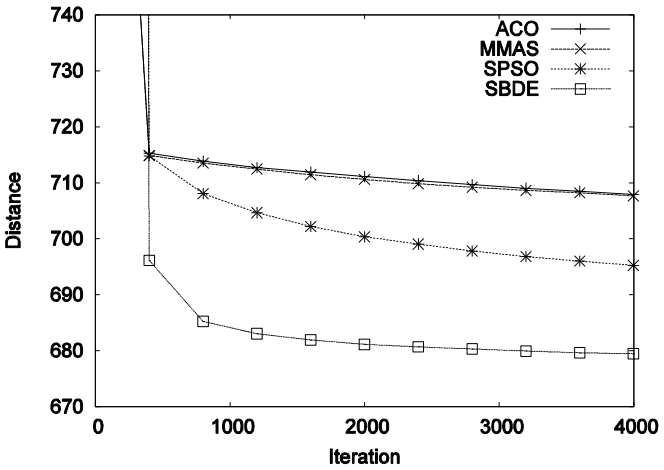

Fig.3 Results of mean value for st70

obtained by ACO, MMAS, SPSO and SBDE

The optimal value in the case of st70 is 675. The results in Table 3 and Fig. 3 are averaged 1000 runs. The maximum number of iterations in each run is 4000. It can be seen clear that the worst value, the best value and the mean value gained by the proposed algorithm SBDE are the minimum among the methods in Table 3 and the convergence of SBDE is fastest than the others in Fig.3. SBDE is the most effective in quality than the others here and the convergence of SBDE is the fastest.

Table 4 and Fig.4 show, for eil76, the results of the comparison among GA, ACO, MMAS, SPSO, and SBDE.

Table 4 Results for eil76

obtained by GA, ACO, MMAS, SPSO and SBDE

Algorithm	Worst	Best	Mean
GA	2443.71	2015.99	2257.62
ACO	572.50	559.20	569.73
MMAS	570.62	559.64	565.57
SPSO	583.292	552.50	562.68
SBDE	564.10	544.37	552.81

978-1-4799-5955-6/14/$31.00 ©2014 IEEE

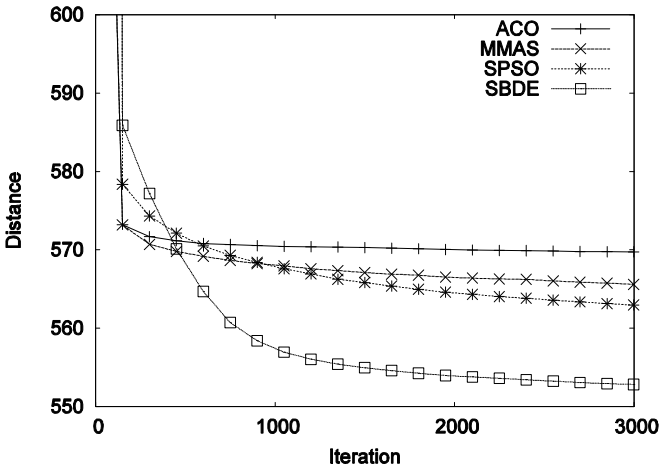

Fig.4 Results of mean value for eil76
obtained by ACO, MMAS, SPSO and SBDE

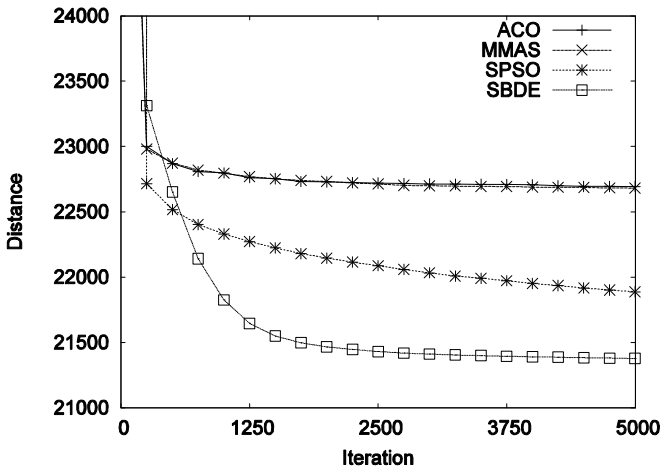

Fig.5 Results of mean value for kraA100
obtained by ACO, MMAS, SPSO and SBDE

The optimal value in the case of eil76 is 538. The results in Table 4 and Fig.4 are averaged 1000 runs. The maximum number of iterations in each run is 3000. It can be seen clear that the worst value, the best value and the mean value gained by the proposed algorithm SBDE are the minimum among the methods in Table 4 and the convergence of SBDE is fastest than the others in Fig.4. SBDE is the most effective in quality than the others here and the convergence of SBDE is the fastest.

Table 5 and Fig.5 show, for kroA100, the results of the comparison among GA, ACO, MMAS, SPSO, and SBDE.

Table 5 Results for kroA100
obtained by GA, ACO, MMAS, SPSO and SBDE

Algorithm	Worst	Best	Mean
GA	170039.0	131037.0	151515.3
ACO	23238.6	22319.0	22693.8
MMAS	23360.7	22288.5	22681.0
SPSO	22613.0	21406.8	21887.9
SBDE	22266.6	21285.4	21377.4

The optimal value in the case of kroA100 is 21282. The results in Table 5 and Fig.5 are averaged 1000 runs. The maximum number of iterations in each run is 5000. It can be seen clear that the worst value, the best value and the mean value gained by the proposed algorithm SBDE are the minimum among the methods in Table 5 and the convergence of SBDE is fastest than the others in Fig.5. The proposed approach is also most effective in quality among these.

Table 6 and Fig.6 show, for eil101, the results of the comparison among GA, ACO, MMAS, SPSO, and SBDE.

Table 6 Results for eil101
obtained by GA, ACO, MMAS, SPSO and SBDE

Algorithm	Worst	Best	Mean
GA	3396.540	2759.810	3091.888
ACO	707.303	672.941	689.890
MMAS	694.830	673.385	689.844
SPSO	710.525	660.192	684.212
SBDE	669.591	640.933	652.428

978-1-4799-5955-6/14/$31.00 ©2014 IEEE

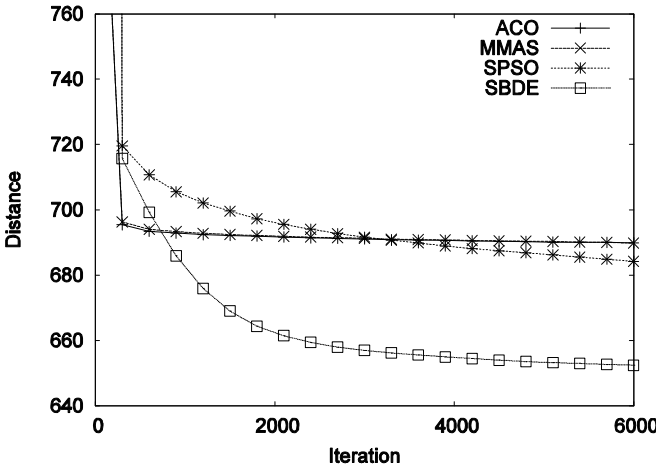

Fig.6 Results of mean value for eil101

obtained by ACO, MMAS, SPSO and SBDE

The optimal value in the case of eil101 is 629. The results in Table 6 and Fig.6 are averaged 1000 runs. The maximum number of iterations in each run is 6000. It can be seen clear that the worst value, the best value and the mean value gained by the proposed algorithm SBDE are the minimum among the methods in Table 6 and the convergence of SBDE is fastest than the others in Fig.6. The proposed approach is also most effective in quality among these.

IV CONCLUSION

In this paper, set-based differential evolution (SBDE) that differential evolution (DE) is based on a set-based representation scheme is proposed. By the numerical experiments for large size TSP that eil51, st70, eil76, kroA100 and eil101, the performance of the proposed approach that SBDE is better than genetic algorithm (GA), ant colony optimization (ACO), max-min ant system (MMAS), set-based particle swarm optimization (SPSO). The convergence of the proposed approach is the fastest than the others. The evolutionary strategies of mutation, crossover and selection can keep the solutions diversity. Also, the accuracy of the proposed approach is GA, ACO, MMAS, and SPSO. The parameters setting can also make influence to the algorithm. The results of the numerical experiments have shown the proposed approach is promising.

REFERENCES

[1] T. Back: Evolutionary Algorithms in Theory and Practice, Oxford University Press, Oxford, 1996.

[2] D. B. Fogel: Evolutionary Computation: Toward a New Philosophy of Machine Intelligence, IEEE Press, Piscataway, NJ, 1995.

[3] K.V.Prince, R.M.Storn, and J.A.Lampinen: Differential Evolution-A Practical Approach to Global Optimization, Springer, 1998.

[4] Tao Liu and Michiharu Maeda, "Set-Based Differential Evolution for Travelling Salesman Problem", in *Proc*. International Conference on Intelligent Networks and Intelligent Systems, pp107-110, 2013.

[5] Applegate, D. L.; Bixby, R. M.; Chvátal, V.; Cook, W. J., *The Traveling Salesman Problem*, ISBN 0-691-12993-2., 2006.

[6] Zhou Yanping, Gu Xingsheng, "Development of Differential Evolution Algorithm," Control and Instruments in Chemical Industry,Vol.34, No.3, pp. 1-5, 2007, doi:1000-3932 (2007) 03-0001-05.

[7] Chakraborty, U. K. ed.: Advances in Differential Evolution, Springer (2008).

[8] Feoktistov, V: *Differential Evolution: In Search of Solutions*. Springer. ISBN 978-0-387-36895-5, 2006.

[9] Das, S. and Suganthan, P., "Differential Evolution: A Survey of the State-of-the-Art", IEEE Transactions on Evolutionary Computation, Vol. 15, No. 1, pp. 4, 2011.

[10] Wei-Neng Chen, et al. "A Novel Set-Based Particle Swarm Optimization Method for Discrete Optimization Problems". IEEE Trans. Evol. Comput., vol.14, no.2, pp.278-299, 2010.

[11] S. Lin and B. W. Kernighan, " An Effective Heuristic Algorithm for the Traveling-salesman problem", Operations Res., vol. 21, no. 2, pp.498-516, Mar.-Apr. 1973.

[12] B. Golden and W. Stewart, "Empiric analysis of heuristics," in *The Traveling Salesman Problem*, E. L. Lawler, J. K. Lenstra, A. H. G.Rinnooy-Kan, and D. B. Shmoys, Eds. New York: Wiley, 1985.

[13] G. Reinelt, "TSPLIB: A Traveling Salesman Problem Library", Orbit Reconst. Simulation Anal. J. Comput., vol. 3, no. 4, pp. 376-384, Jan. 1991.

[14] Yuhui Shi and Russell Eberhart, "A modified particle swarm optimizer," in *Pro. IEEE Int. Conf. Evol. Comput.*, pp. 69-73, 1998.

978-1-4799-5955-6/14/$31.00 ©2014 IEEE