MATEC Web of Conferences **44**, 02025 (2016)
DOI: 10.1051/matecconf/20164402025
© Owned by the authors, published by EDP Sciences, 2016

Double evolutional artificial bee colony algorithm for multiple traveling salesman problem

Ming Hao Xue[1], Tie Zhu Wang[2] and Sheng Mao[3]

[1] *Materiel Management & Safety Engineering College, Air Force Engineering University,* Xi'an 710043, *China*
[2]*Information Management Centre, Air Force Engineering University,* Xi'an 710043, *China*
[3] *Aeronautics and Astronautics Engineering College, Air Force Engineering University,* Xi'an 710043, *China*

Abstract: The double evolutional artificial bee colony algorithm (DEABC) is proposed for solving the single depot multiple traveling salesman problem (MTSP). The proposed DEABC algorithm, which takes advantage of the strength of the upgraded operators, is characterized by its guidance in exploitation search and diversity in exploration search. The double evolutional process for exploitation search is composed of two phases of half stochastic optimal search, and the diversity generating operator for exploration search is used for solutions which cannot be improved after limited times. The computational results demonstrated the superiority of our algorithm over previous state-of-the-art methods.

Key words: double evolutional artificial bee colony algorithm; optimization; multiple travelling salesman problem

1 Introduction

The multiple travelling salesman problem (MTSP) is a generalization of traveling salesman problem (TSP), which is a well-known NP-hard problem [1]. A new variation of artificial bee colony (ABC) algorithm, called double evolutional artificial bee colony (DEABC) algorithm is presented by taking advantage of the strength of different operations to improve the exploitation search ability and increase the convergence speed when solving MTSP. The current local search operators are characterized by different variation they can cause to the original solution, but a single operation has actually been wasted most of the time, because the process can neither create a better solution nor generate the best solution it can possibly search.

Evolutionary algorithms (EAs) and Swarm Intelligence (SI) based algorithms are two major branches of population-based optimization algorithms [2]. The ABC algorithm is one of most recently population-based methods [3] which imitates the intelligent behaviour of honey bee swarm for seeking nectar. The robust search process consists of exploration search and exploitation search, the former aims at finding promising new solutions, and the latter is to find the optimum in the neighbourhood of a good solution [4]. However, ABC is good at exploration but poor at exploitation and its convergence speed is also an issue in some cases [5,6]. Moreover, literature addressing ABC algorithm and MTSP is quite rare.

The ABC algorithm is a simulation of the swarm intelligence of honey bees for searching nectar. The artificial bee colony includes three kinds of artificial bees: employed bees, onlooker bees and scout bees. Each kind of bees corresponds to a phase in this algorithm.

After initialization, each employed bee is associated to a known food source. In employed bee phase, each employed bee performs searching in the neighborhood of its current food source, and the better ones are reserved according to fitness. In onlooker bee phase, onlooker bees first follow employed bees proportionally to the probability value, which is determined by the fitness of the food source; and then seeking a solution in the neighborhood of the food source found by the followed employed bee; finally, a better food source is reserved. In the scout bee phase, a scout bee searches a food source without the help of colony's memory.

In general, the number of employed bees and onlooker bees are equal to half of the colony size. An employed bee becomes a scout bee when its food source is abandoned; the abandonment happens when the food source cannot be improved after limited times of searching (the food source has been fully exploited).

In ABC algorithm each food source represents a possible solution for the problem and the fitness value of the solution corresponds to the nectar amount of this food source. Basic steps of the ABC algorithm are given below:

1. Initialization: randomly generate a set of solutions; a solution represents a currently known food source, and each employed bee is assigned to a food source.

2. Calculate and memorize the fitness of each food source:

[1] Corresponding author: qweilm@163.com

$$fitness_i = \begin{cases} \frac{1}{fit_i}, fit_i \geqslant 0 \\ 1 + abs(fit_i), fit_i < 0 \end{cases} \quad (1)$$

where fit_i is the cost value (value of objective function),

3. For each employed bee:

a. Perform local search in the neighborhood of her assigned food source

b. Calculate the fitness of the detected food source

c. Memorize the better food source

4. For each onlooker bee:

a. Choose a food source from the current found ones depending on the roulette wheel using the probability p_i , which is calculated as follows:

$$p_i = \frac{fitness_i}{\sum_{j=1}^{SN} fitness_i} \quad (2)$$

where *SN* is the number of solutions

b. Repeat the searching process in step 3

5. If a food source hasn't been improved after limited times, the corresponding employed bee becomes a scout bee and randomly generate a solution.

6. Repeat the step 3-6 until the maximum cycle times are completed

7. Output the optimal result.

2 Improvements for local search operators

The operators can be upgraded for two objectives: one is to obtain guidance by optimal searching mechanism; the other is to obtain greater diversity by expanding searching space. The former has an advantage over exploitation search and convergence speed, the latter can be used to further improve exploration search.

2.1 Half-stochastic optimal searching operators

Using the operators which are inspired by [7] for local search, the operations can be actually wasted most of the time, because a single searching process can neither create a better solution nor generate the best solution it can possibly search. For the first objective, we propose a new searching method, which aims to find the optimal solution in the searching space of an operator.

Taking the swap points operation as an example, the optimal searching strategy is to calculate the fitness for all possible swaps, finally swap the position of a pair of cities with the optimal fitness. However, the searching space of swap points is C_n^2, the convergence speed would be greatly compromised if we want to obtain the optimal solution.

After comprehensively considering the value of coefficient α , optimal searching space and convergence speed, an approach called half-stochastic optimal searching is proposed, and only three initial operators are qualified to be modified, they are swap points, insert point and reverse subsequence.

The procedures for half-stochastic optimal swap points (insert point or reverse subsequence) operator are as follows:

1.Generate a positive integer *r* randomly, the city in position *r* acts as a swap point (insert point or an edge of a subsequence), the value of *r* can't exceed *n*.

2.Calculate the fitness for generated solutions of all possible swaps of points (all insert positions or all subsequences can be reversed).

3.Reserve the optimal solution.

Different values of α can affect their local search ability, but they are the ones can be realized by the idea of half-stochastic optimal searching.

Obviously, the searching range can not only be the whole sequence, but also one or several tours, which is suitable to be modified for different optimizing objectives.

2.2 The diversity generating operators

The second upgrading objective takes advantage of operators with bigger value of α , which can generate new solutions with greater diversity. The α value of swap points, swap subsequences and swap-reverse subsequences are all 4, and they can all be suitable for this generating diversity.

The operation of swap can be put into two categories: one is the operated points (sequences) are within the same tour, another are located in different tours. They are named after inner-tour swap and cross-tour swap respectively. We set the times to perform the two kinds of swap operations in linear correlation with *m* (the number of salesman); so the need for more diversified solutions can be met when *m* varies.

The procedures for diversity generating operator are as follows:

1.Generate a positive integer *r1* randomly, where the value of *r1* can't exceed *m-1*(*m* is the number of salesman); define *r2=m-r1*.

2.Perform *r1* times of inner-tour swap operations; the tour is chosen randomly each time

3.Perform *r2* times of cross-tour swap operations; the two tours and the cities are chosen randomly each time

This sort of operators can be effective when a solution doesn't improve after certain times of iteration (a food source is fully exploited).

3 Double evolutional artificial bee colony algorithm

This section proposes a double evolutional artificial bee colony algorithm (DEABC) for solving the single depot multiple traveling salesman problem (MTSP).

3.1 Initialization

Firstly, we send all bees to find food sources, then the 50% bees finding the better solutions become employed bees; the rest become onlooker bees and return to the hive. Food sources are evaluated by their fitness. The number of employed bees and onlooker bees are the same, both equal to the half of the colony size.

The process to find food sources generally consists of two procedures: firstly, randomly generate a permutation of *n* cities; secondly, randomly divide the sequence into *m* tours. The *m* is the number of salesman, and at least one city is supposed to be included in a tour.

3.2 Fitness

We adopt the function (1) to calculate fitness, and the cost function used is the same as the objective function. Since the two classic test objectives are minimizing the total distance (objective 1) and minimizing the longest tour (objective 2), we have two different cost functions. We need to maximize the fitness in both cases.

3.3 Selection

After the employed bee phase, the onlooker bees choose the food sources found by the employed bees based on probability p_i expressed as follows:

$$p_i = \frac{fitness_i}{\sum_{j=1}^{SN} fitness_i} \tag{3}$$

Where $fitness_i$ is the fitness of the food source *i*.

3.4 The double evolutional process

The key to this algorithm is the double evolutional process, which is specially designed for exploitation search. The exploitation search is performed in both employed bee phase and onlooker bee phase, and two evolutional approaches are proposed to enhance the search ability. For both objectives, the insert point operator and reverse subsequence operator are adopted and modified to constitute the process. The double evolutional procedures are as follows:

1.Conduct the half-stochastic optimal reverse subsequence operator for the whole sequence

2.Calculate the fitness of the generated solution and reserve the better one

3.Conduct the half-stochastic optimal insert point operator for the whole sequence generated in step 2.

4.Calculate the fitness of the generated solution and reserve the better one

The procedures of the operators can be referred to in section 2.1. The half-stochastic optimal reverse subsequence operator has the minimum α value, which has an edge over convergence, calculation and selection; it also has minimum searching space, which further increases the convergence speed. The α value of half-stochastic optimal insert operator is medium, but its searching space is minimum, which enhances the exploitation searching ability in a greater range.

This process is fit for both objectives, because the results of double evolutional process turn out to be fitness-oriented.

3.5 The exploration search

If a food source hasn't been updated after certain times (limit) of exploitation search, it will be abandoned and replaced by a new solution. The new solution is found around the abandoned solution and implemented by the diversity generating operator described in 2.2.

For objective 1, the swap subsequence operation is applied to generate a new solution; for objective 2, we choose a city randomly from a tour other than the longest tour and swap it with the city in longest tour which obtains the minimizing longest tour. For both objectives, we reserve the current optimal solution in case it is abandoned.

The number of scout bee in an iteration is set as 1, and the number of swap operation in an exploration search is set as 1. The reason for not increase the number of them is the same results can be obtained by decreasing the value of limit.

The framework of DEABC is illustrated in figure 1.

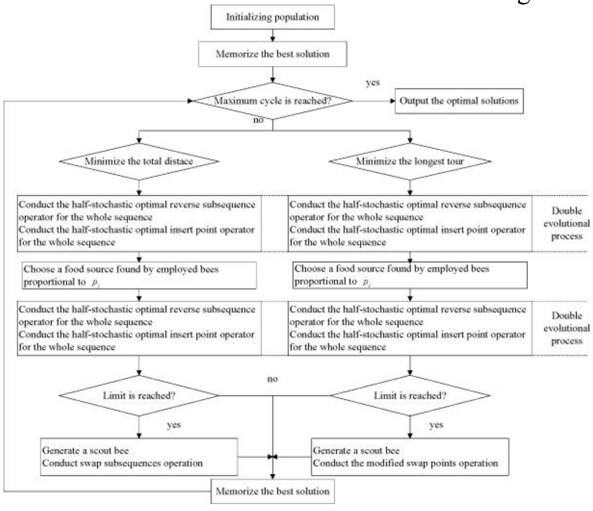
Initializing population
Memorize the best solution
Maximum cycle is reached?
yes
Output the optimal solutions
no
Minimize the total distace
Minimize the longest tour
Conduct the half-stochastic optimal reverse subsequence operator for the whole sequence
Conduct the half-stochastic optimal insert point operator for the whole sequence
Double evolutional process
Choose a food source found by employed bees proportional to p_i
Limit is reached?
no
yes
Generate a scout bee
Conduct swap subsequences operation
Conduct the modified swap points operation
Memorize the best solution

Figure.1 The framework of DEABC

4 Computational results

To evaluate the performance of DEABC, test problems in [8] is used to compare the computational results. The data are Euclidean, two dimensional symmetric problems, and the first city is always used as depot. The experimental instances with different problem size and salesmen combinations are listed in table 1. For the problem, the two objectives are optimized separately.

The results of previous algorithms are also shown for comparison. The genetic algorithm with one chromosome representation and the genetic algorithm with two chromosomes used in [8]; the two-part chromosome representation based on genetic algorithm proposed in [8] is referred to as GA2PC, the steady state grouping genetic algorithm proposed in [9] is referred to as GGA-SS; the ant colony algorithm proposed in [10] is referred to as ACO.

Table.1 Benchmark instances

Name	Cities (n)	Salesmen (m)	Origin
MTSP-1-51	51	3,5,10	[8]

MTSP-1-100	100	3,5,10,20	[8]
MTSP-1-150	150	3,5,10,20,30	[8]

The values of parameters are shown in the table 2, which are chosen empirically.

Table.2 Parameter settings in DEABC

parameters	value
Colony size	100
Maximum cycle number	1000
Limit	5
Number of employed bees	50
Number of onlooker bees	50
Number of scout bees	1

4.1 Comparison of proposed operators

First of all, a test problem is chosen for several operators proposed in this paper to compare their performance. The improved operators can be put to two categories--half stochastic optimal searching and generating diversity, we run this test because no theoretically prove has been given to determine their capabilities.

The test problem is to reveal the abilities of different operators, and prove the rationality of the design of the double evolutional process for minimizing total distance. The double evolutional process for minimizing longest tour is the same, and the exploration search for it is especially designed, so we only run test for the first objective.

We chose the MTSP-1-51 as test data, and the number of salesman is set as 3, 30 independent runs are tested for each different combination. The results are shown in table 3, where "1" represents the half-stochastic optimal swap points operator, "2" means the half-stochastic insert point operator, "3" represents the half-stochastic reverse subsequence operator, and the short line connecting different numbers means the combination of the operators; S.D. stands for standard deviation of objective values of 30 runs, time is the average time cost in seconds for each run. Considering the scale of test data, the diversity generating operators only conduct swap operation for one time in each scout bee phase.

Table.3 Experimental results of different operators

operator		1	2	3	1-2	1-3	2-3	1-2-3
Swap points	Max	580	556	471	511	466	454	454
	Min	517	468	446	452	447	446	446
	Mean	548	514	457	478	456	450	450
	S.D.	17	23	6	14	5	2	2
	Time	2.07	2.30	2.63	4.03	4.40	4.67	6.51
Swap subse-quence	Max	613	573	474	493	462	453	454
	Min	510	456	452	449	448	446	446
	Mean	545	517	459	471	454	450	450
	S.D.	20	23	5	10	4	2	2
	Time (s)	2.12	2.37	2.71	4.07	4.50	4.67	6.40
Swap-reverse	Max	580	532	465	494	458	454	454
	Min	504	463	447	452	447	446	446
	Mean	534	495	458	471	454	449	450
	S.D.	21	21	5	12	3	2	2
	Time	2.07	2.33	2.67	4.07	4.40	4.67	6.43

As can be concluded from table 3, for half-stochastic optimal searching operators, the reverse operator works best, while the swap operator **is** the worst; when combining the half-stochastic optimal insert and reverse operators, the results are the best; when combining all the operators, the results aren't been improved, but the time cost is the most. Based on the fact that the half-stochastic optimal reverse and insert operators can be fully capable of obtaining ideal results, the double evolutional process for minimizing total distance are designed to be the combination of the two operators.

For diversity generating operators, the swap subsequence operator works best with the double evolutional process, thus been chosen as the operator to perform exploration search.

4.2 Comparison of approaches on test problems in [8]

Table 4 and 5 compare the performance of DEABC with GA1C, GA2C, GA2PC, GGA-SS, ACO and TCX of average solution quality for objective 1 and objective2 on test problems used in [8]. The results of GA1C, GA2C, GA2PC are taken from [8], the results of GGA-SS, ACO and TCX are taken from [9], [10]. The S.D. is short for standard deviation of objective values.

As can be seen from table 4, the DEABC can obtain the best results for test problems1-7and 9-12, but it doesn't outperform ACO and GGA-SS on the 8th test problem.

For minimizing the longest tour, the superiority of DEABC is not as obvious as that of objective 1. Basically, it works well on small scale test data or the test data with more salesmen. When the number of city is relative big and the number of salesmen is relatively few, the results cannot outperform ACO and GGA-SS.

Table.4 Results for optimizing the total distance on test problems from [8]

Instances			GA1C	GA2C	GA2PC	GGA-SS	ACO	DEABC		
	n	m						Mean	S.D.	Best
1	51	3	529	570	543	449	448	450	2	446
2	51	5	564	627	586	479	478	477	3	471
3	51	10	801	879	723	584	584	584	3	579
4	100	3	27036	30972	26653	22051	22619	22979	530	21878
5	100	5	29753	44062	30408	23678	24166	24433	454	23666
6	100	10	36890	65116	31227	28488	27890	28110	565	27375
7	100	20	62471	95568	54700	40892	39949	39304	400	38377
8	150	3	46111	48108	47418	38434	39247	39900	344	39283
9	150	5	49443	51101	49947	39962	40647	40643	430	40032
10	150	10	59341	64893	54958	44274	44436	44213	420	43464
11	150	20	94291	100037	73934	56412	55980	55605	545	54617
12	150	30	131503	143476	99547	72783	71266	70824	497	69433

Table.5 Results for minimizing the longest tour on test problems from [8]

Instances			GA1C	GA2C	GA2PC	GGA-SS	ACO	DEABC		
	n	m						Mean	S.D.	Best
1	51	3	234	275	203	161	160	168	5	160
2	51	5	173	220	164	119	118	125	3	119
3	51	10	140	165	123	112	108	112	0	112
4	100	3	14722	16229	13556	8542	8817	9337	412	8755
5	100	5	11193	11606	10589	6852	6964	7561	397	7056
6	100	10	9960	10200	9463	6370	6363	6482	62	6363
7	100	20	9235	9470	8388	6359	6356	6358	0	6358
8	15	3	19875	21067	19687	1326	1388	1490	786	1372

	0					8	5	5		2
9	150	5	15229	15450	14748	8660	9270	10203	508	9553
10	150	10	12154	12382	11158	5875	6132	6721	229	6232
11	150	20	10206	10338	10044	5252	5250	5387	94	5246
12	150	30	9626	9683	8775	5247	5246	5246	0	5246

5 Conclusions

This paper proposed a double evolutional artificial bee colony algorithm for solving the MTSP. The operators are then modified as half-stochastic optimal searching operators and diversity generating operators, which lay the foundation for the double evolutional process and the exploration search. Further improvements are made to the operators for different optimizing objectives. The computational results showed the competitiveness of DEABC for other algorithms.

The future work can be focused on three aspects. The first is to improve the searching ability over large scale test data. DEABC cannot outperform ACO and GGA-SS over a few test problems, so we wish to improve the algorithm by dividing the large scale test data into several relatively groups, then establish a communication mechanism to obtain better results. The second is to develop the special operators for minimizing the longest tour of all salesmen. The third is to extend the DEABC to solve MTSP with multiple depots, because the situation with multiple depots can be applied to a wider range of practical situations.

References

[1] Gorenstein S. Printing press scheduling for multi-edition periodicals. Management Science 1970;16(6):B373-83.

[2] Szeto,W., Wu,Y., &Ho,S.C.(2011). An artificial bee colony algorithm for the capacitated vehicle routing problem. European Journal of Operational Research,215(1),126-135.

[3] D. Karaboga, B. Akay. A comparative study of Artificial Bee Colony algorithm. Appl. Math. Comput.214 (2009) 108-132.

[4] Karaboga, D. (2005). An idea based on honey bee swarm for numerical optimization. Technical report-tr06, Erciyes university, engineering faculty, computer engineering department.

[5] B.Akay, D.Karaboga, A modified Artificial Bee Colony (ABC) algorithm for real-parameter optimization, Inf. Sci 192 (2012) 120-142.

[6] W. Xiang, M. An. An efficient and robust Artificial Bee Colony algorithm for numerical optimization. Comput. Oper. Res. 40 (2013) 1256-1265.

[7] S. Biswas, S. Das, S. Debchoudhury, S. Kundu, Co-evolving bee colonies by forager migration: A multi-swarm based Artificial Bee Colony algorithm for global search space, Appl. Math. Comput. 232 (2014) 216-234.

[8] A. E. Carter, C. T. Ragsdale, A new approach to solving the multiple traveling salesperson problem using genetic algorithms, Eur. J. Oper. Res. 175 (2006)245-257.

[9] A.Singh,A.S. Baghel, A new grouping genetic algorithm approach to the multiple traveling salesperson problem, Soft Comput. 13 (2009)95-101.

[10] W. Liu, S. Li, F. Zhao, A. Zheng, An ant colony optimization algorithm for the multiple traveling salesmen problem, in: 4th IEEE Industrial Conference on Industrial Electronics and Applications (ICIEA 2009), IEEE, 2009, pp. 1533–1537.