The Tenth International Symposium on Operations Research and Its Applications (ISORA 2011)
Dunhuang, China, August 28–31, 2011

An Improved Artificial Bee Colony Algorithm for the Capacitated Vehicle Routing Problem with Time-dependent Travel Times

Ping Ji[1] Yongzhong Wu[1,2]

1 Department of Industrial and Systems Engineering, The Hong Kong Polytechnic University, Hong Kong
2 School of Business Administration, South China University of Technology, Guangzhou, P.R., China

Abstract The capacitated vehicle routing problem with time-dependent travel times has many real-life applications. Due to the computational complexity, metaheuristics are commonly used to solve the problem. In this paper, a relatively new metaheuristic, Artificial Bee Colony (ABC) algorithm, is developed to solve the problem. Besides, the ABC algorithm is revised to improve its performance. Benchmark problem instances of different sizes are generated and used for experimental tests. The results show that the revised ABC can significantly improve the performance of the algorithm in solving the problems.

Keywords Operations Research, Vehicle Routing Problems, Artificial Bee Colony

1 Introduction

The vehicle routing problem (VRP) has received much interest among researchers due to its widespread applications. The basic VRP models assume constant travel times between locations (customers). This assumption is not realistic, particularly in urban areas. In a congested urban environment, the travel time between customers is usually not a function of distance traveled alone since speeds are not constant. For example, the traveling times during the morning and evening rush hours are greater than those in other time periods of the day. If the major variation in travel times results from the time-of-day variation, the travel time between two customers may be represented by a deterministic function of the distance between the two points and also the time of day the travel takes place [1]. Considering the time-dependent property in solving the vehicle routing problem could be beneficial for improving the quality of the routing plan.

Nevertheless, the time-dependent property of travel times may significantly increase the computational complexity of the traditional vehicle routing problem, which is already *NP*-hard. Highly efficient heuristics or metaheuristic methods are required for obtaining near-optimal solutions. In the literature, well-established metaheuristics, like genetic algorithms, have been proposed to solve the problem. In this paper, a newly-developed metaheuristic method, the Artificial Bee Colony

algorithm (ABC) is adopted for solving the problem. In addition, the original scheme for the ABC is revised in order to improve the performance of the algorithm. A set of benchmark problem instances of different sizes are developed and used for experimental tests. The remaining part of this paper is organized as follows. Section 2 reviews the related literature. In Section 3, the general artificial bee colony algorithm is presented while Section 4 describes how to adopt the ABC algorithm to the vehicle routing problem. The ABC algorithm is revised in Section 5 and experimental tests and analysis are conducted in Section 6. Finally, the paper is concluded in Section 7.

2 Literature Review

The vehicle routing problem (VRP) is one of the classic combinatorial optimization problems, which is *NP*-hard. Even for the traditional vehicle routing problem, only small instances can be solved to optimality by exact methods. Therefore, heuristic or meta-heuristic methods are commonly used in practice for solving the vehicle routing problem [2].

Comparatively, only a few researchers have considered the time-dependent travel times in the vehicle routing problem. Malandraki and Daskin [1] considered the so-called Time Dependent Vehicle Routing Problem (TDVRP). Hill and Benton [3] also studied the vehicle routing problem with time-dependent travel times and proposed a simple greedy heuristic for the problem. The time-dependent travel times associated with long-term forecasts were addressed by Ichoua et al. [4]. They proposed a parallel tabu search for solving the problem. Hashimoto et al. [5] proposed an iterated local search algorithm for the vehicle routing problem with time-dependent travel times and time windows.

The Artificial Bee Colony algorithm (ABC) is a new population-based metaheuristic approach proposed by Karaboga [7]. This approach is inspired by the intelligent foraging behavior of honeybee swarm. Karaboga and Basturk [8] demonstrated that ABC outperforms genetic algorithms and particle swarm optimization in multivariable function optimization. Singh [9] compared ABC against three other metaheuristic approaches, including a genetic algorithm, an ant-colony optimization approach and a tabu search approach, on solving the leaf-constrained minimum spanning tree problem. He showed that the ABC outperforms all the other approaches, except in one instance, where the average solution of the tabu search is better.

The consideration of time-dependent property of travel times significantly deteriorates the adaptation of VRP algorithms [6]. Highly effective and efficient approaches are required for solving the vehicle routing problem with time-dependent travel times. In this paper, an artificial bee colony algorithm is proposed for solving the vehicle routing problem with time-dependent travel times.

3 The artificial bee colony (ABC) algorithm

In an ABC algorithm, the colony of bees is divided into employed bees, onlookers and scouts. Employed bees are responsible for exploiting available food sources

(solutions) and gathering required information. These bees also share information with onlookers, and onlookers select a found source near the food sources chosen by the employed bees. When the source is abandoned, the employed bee becomes a scout and starts to search a new source in the vicinity of the hive. This abandonment happens when the quality of the food source is not improved after performing a maximum allowable number of cycles, called limit.

The scheme of the ABC algorithm can be described as follows.

Scheme of the ABC

Step 1. Randomly generate a set of solutions as initial food sources. Assign an employed bee to each food source

Step 2. Evaluate the nectar amount (fitness) of each food source

Step 3. **In each iteration**

- 3.1 Apply the neighborhood operator to each food source to generate a neighbor food source. If the fitness of the neighbor food source is better than that of the original food source, then replace the original food source with this neighbor food source
- 3.2 For each onlooker, use the fitness-based roulette wheel selection method to select a food source, say *A*. Apply a neighborhood operator to *A* to find a neighbor food source, say *A*'
- 3.3 For each food source *A*, if the fitness of the best one among all *A*'s is better than that of *A*, then use this best one to replace *A*
- 3.4 Replace the food sources whose fitness has not been improved for *limit* iterations with randomly generated solutions
- 3.5 Go to next iteration until the predetermined stopping criterion is met

Step 4. Output the best food source (solution) found

4 The ABC for the problem

The investigated capacitated vehicle routing problem can be described as follows. Several (say, *k*) vehicles, stationed in a depot, are required to collect (or deliver) goods from many (say, *n*) customers with various demands. The capacitated VRP consists of designing a set of at most *k* collection (or delivery) routes such that (1) each route starts and ends at the depot; (2) each customer is visited exactly once by exactly one vehicle; (3) the total demands of customers in each route does not exceed the vehicle capacity, such as weight or volume; and (4) travel times on each link are time-dependent; (5) total routing cost (or time) is minimized.

In this paper, an ABC is proposed for solving the capacitated vehicle routing problem (CVRP) with time-dependent travel times. In the following, the algorithm is described by discussions on the representation scheme, initialization method, neighborhood operators, constraint handling methods, and selection method of the food sources.

4.1 Representation scheme and initialization

The solution to the problem consists of two stages. The first stage is the clustering

of customers, and the other is the routing for each cluster. In order to maintain the simplicity of the ABC algorithm, a straightforward representation scheme is adopted. Both the clustering and routing stages are represented in a single string. Figure 1 illustrates a representation for a CVRP instance with $n = 7$ and $k = 3$. As shown in Figure 1, customers 4 and 1 are assigned to the same vehicle (the first vehicle), and the routing sequence is the same as the customer order (customer 4 first, then customer 1 for vehicle 1). If two adjacent numbers are 0, then this vehicle is not used.

0	4	1	0	2	7	5	0	3	6

Figure 1. Representation scheme

4.2 Constraint handling method

Due to the capacity constraints, infeasible solutions may cover a big proportion of the solution space. We allow the search to be conducted in the infeasible part of the solution space. The constraints of the problem include the maximum loading capacity of each vehicle and the maximum traveling duration for each vehicle. A dynamic penalty method is used. That is, when evaluating a solution, the violation on loading capacity and maximum traveling duration will be treated as a penalty to the original objective value, which is the total travel time. For a solution x, let $c(x)$ denote its total travel time, and let $q(x)$ and $t(x)$ denote the total violation of the loading demand and duration constraints, respectively. Each solution is evaluated by a cost function $z(x) = c(x) + aq(x) + bt(x)$, where a and b are self-adjusting positive parameters. Parameter a is adjusted as follows: in each iteration, if the number of food sources with no loading constraint violation is greater than half of the total number of food sources, the value of a is divided by $1 + \delta$, otherwise it is multiplied by $1 + \delta$, where δ is a positive parameter. A similar rule applies to b with respect to the route duration constraint.

4.3 Neighborhood operators

In each iteration of the ABC, each onlooker selects a food source and uses a neighborhood operator to find a neighboring solution. The selection of the food source is fitness-proportional. In the proposed algorithm, the common roulette wheel selection method is used. Neighborhood operators are used by both the employed bees and the onlookers to find new solutions around the food sources. For the proposed ABC, three neighborhood operators are tried, i.e. swap, insert, and inverse. The performance of these operators will be compared. The swap operator randomly selects two points in the solution and swaps their positions. The insert operator randomly selects one point in the solution and moves it to another position. The inverse operator works by randomly selecting a section of continuous points and inversing the sequence of these points.

5 A revised ABC

Despite of specific operators and fine-tuned parameters, preliminary experiments show that the overall performance of the original ABC on solving the problem is only moderate. Therefore, the scheme for the artificial bee colony algorithm is

revised to improve the performance of the algorithm.

The new scheme for the ABC is derived from two assumptions from the behavior of the bee colony. First, we suppose that any new place found by the onlookers around a certain food source will become a new food source to compete with all other food sources, rather than simply replace the old food source. Second, it is assumed that after a food source is abandoned, the concerned employed bee will find a new food source in the vicinity of the abandoned food source.

Based on the above two assumptions, two revisions are made to the ABC algorithm. First, the best solution found by the onlookers in vicinity of a certain food source will **NOT** replace this food source, opposite to the case in the original ABC scheme. Instead, the new solution will replace the food source which has not been improved for the greatest number of iterations among those whose solutions are worse than the concerned new solution. In this way, a potential food source will lead to more potential food sources in the vicinity of it. This may be beneficial in that more search branches will be developed from a potential node at any stage of the algorithm. Secondly, a food source which has not been improved for the defined number of iterations will **NOT** be replaced by a randomly generated solution. Instead, the neighborhood operator will be applied to obtain a new solution in the vicinity of the abandoned food source, no matter the new solution is better than the abandoned food source or not. The second revision may be beneficial because it prevents the algorithm from starting with totally new solutions and makes most benefits out of the current solutions.

6 Experimental Tests

In this paper, a set of problem instances were generated. The revised ABC algorithms were tested on the generated instances.

6.1 Generation of testing instances

A set of problem instances were generated with different numbers of customers, i.e., 20, 50, 100, 150, 200, 250 and 300. For each instance, the customer locations were randomly distributed within a square area of 100 kilometers × 100 kilometers, and the numbers of vehicles were set to be 1/10 of the number of customers. The demands for customers were randomly generated from a uniform [1, 20] distribution with the result rounded to an integer, and the service durations were generated from a uniform [15 minutes, 30 minutes] distribution with the result rounded to an integer. The capacity of each vehicle, Q, in each instance was set as follows:

$$Q = c \cdot \sum_{i=2}^{n} d_i / k \tag{1}$$

where c is the ratio of total capacity to total demand. The value of c is randomly generated from (1.1, 2.0), while a smaller value represents a higher degree of capacity constraint. d_i is the demand of customer i.

The whole planning horizon was set to be 12 hours, divided into 6 equal periods, with 2 hours for each period. In order to create different degree of time-dependency,

four uniform distributions with range [0.5, 0.5], [0.4, 0.6], [0.3, 0.7], [0.2, 0.8] were used to generate the traveling speeds (kilometers per minute) for each link during each period. These four distributions have the same means but different ranges, while the range [0.5, 0.5] represents no time-dependency of the travel times. For the period after the 12-hour planning horizon, the speeds were set to a very small value so that all vehicles are forced to complete the services before the end of the planning horizon. Totally 28 instances were generated.

6.2 Experiments

The algorithms were coded in Visual C++ 2003, and ran on a computer with 1.73 GHz and 1G RAM. For the experiments, some common parameters were set as follows: The numbers of the employed bees and the onlookers were both 25. The initial values for both a and b were 0.1. The value of δ was set to be 0.001.

6.2.1 Comparison of neighborhood operators

Three duplicates of the ABC algorithm, each with a neighborhood operator were implemented. The three algorithms were tested on the instance tdvrp28 which has 300 customers. Each of the three algorithm duplicates were run 20 times, and each run terminated after 30,000 iterations were made. It was found that among the three neighborhood operators, the swap operator achieves the best average objective value, while the insert operator does not yield satisfactory results. The invert mutation seems to have close performance as the swap, but with greater computational time. Since no evidence shows that combination of these three operators can yield better results, the swap operator is used only in the revised ABC algorithm.

6.2.2 Determination of parameter *limit*

In the ABC, if a food source cannot be improved during a predetermined number of iterations, then this food source will be abandoned. This is done by producing a position randomly and replacing it with the abandoned one. The value of predetermined number of iterations is an important control parameter of the ABC, which is called "*limit*" for abandonment. We examined the effect of the parameter *limit* on the performance of ABC on solving the investigated problem. Through trial and error, it was found that *limit* = 80*n* is suitable for the problem.

6.2.3 Comparison of old scheme and new scheme

Two duplicates of the ABC were implemented, one using the original scheme (old ABC) and the other using the new scheme (revised ABC). The two algorithms were tested on all the instances. For each instance, each ABC runs 20 times. For each run, the number of iterations was set to be 500*n*.

Table 1 shows the computational results. For all the 28 instances, the ABC with the new revised scheme obtains better solutions than the old ABC. The average percentage improvement for all the 28 instances is 3.25%. In addition, for 24 out of all 28 instances, the standard deviation of the results obtained by the revised ABC is smaller than that by the old scheme, indicating that the revised ABC is more robust. The revised ABC is also more efficient in terms of CPU times. These results show that the revised scheme can significantly improve the performance of the ABC in terms of both effectiveness and efficiency.

Table 1: Computational results for old and revised ABC

Inst.	Old ABC[a]				New ABC[b]			
	best[c]	avg.[d]	s.d.[e]	CPU[f]	best[c]	avg.[d]	s.d.[e]	CPU[f]
tdvrp01	886.49	886	8.78	0.2	886.49	886	6.56	0.2
tdvrp02	861.61	870	11.61	0.2	861.61	862	9.08	0.2
tdvrp03	835.28	851	18.55	0.2	829.26	845	19.02	0.2
tdvrp04	832.79	847	17.25	0.2	817.42	831	16.07	0.2
tdvrp05	1418.76	1492	65.02	0.7	1359.02	1430	72.14	0.7
tdvrp06	1322.66	1382	59.43	0.7	1304.87	1359	45.67	0.7
tdvrp07	1315.75	1353	33.08	0.7	1277.39	1312	27.91	0.7
tdvrp08	1262.79	1348	79.29	0.7	1221.90	1309	68.39	0.7
tdvrp09	2467.79	2531	65.98	2.5	2382.26	2443	61.44	2.2
tdvrp10	2306.57	2528	266.85	2.5	2289.61	2495	120.40	2.3
tdvrp11	2345.09	2502	201.53	2.5	2296.22	2441	131.67	2.3
tdvrp12	2373.52	2490	99.84	2.7	2287.94	2401	102.76	2.3
tdvrp13	3957.12	4123	119.77	5.7	3886.27	4050	124.38	5.1
tdvrp14	3865.19	4110	129.53	5.7	3811.90	4047	92.21	5.1
tdvrp15	3734.32	3879	107.88	5.8	3656.50	3798	105.16	5.1
tdvrp16	3601.19	3829	100.60	5.8	3513.70	3734	85.96	5.1
tdvrp17	4632.25	4929	335.48	10.3	4535.86	4821	278.47	9.0
tdvrp18	4542.47	5133	682.12	9.9	4398.75	4942	553.25	9.2
tdvrp19	4423.84	4878	618.32	9.9	4312.24	4742	444.45	9.1
tdvrp20	4225.89	4682	635.52	10.0	4105.12	4527	456.34	9.1
tdvrp21	5265.43	5743	643.29	22.3	5076.35	5529	619.13	20.3
tdvrp22	5266.58	5854	783.93	22.9	4971.24	5525	774.26	20.5
tdvrp23	4939.14	5356	455.24	23.7	4824.51	5225	299.03	20.7
tdvrp24	4955.13	5480	761.25	21.4	4834.89	5273	472.38	15.7
tdvrp25	6911.65	7280	471.11	21.3	6639.34	6971	377.94	20.9
tdvrp26	6745.05	7203	800.10	21.6	6508.75	6946	629.01	20.9
tdvrp27	6741.29	7336	691.69	21.6	6513.86	7092	670.51	21.2
tdvrp28	6521.84	6927	596.18	21.8	6143.26	6516	526.39	22.2
average	3519.91	3779	316.40	9.05	3412.38	3655	256.79	8.28

[a] ABC duplicate with original scheme; [b] ABC duplicate with new scheme

[c] Best result obtained for 20 runs; [d] Average result obtained for 20 runs

[e] Standard deviation of results; and [f] Average CPU run time (in min.) for 20 runs

7 Conclusions

In this paper, a recent metaheuristic method, an Artificial Bee Colony (ABC) algorithm has been proposed for solving the capacitated vehicle routing problem with time-dependent travel times. In order to improve the computational performance, a new revised scheme for the ABC algorithm was proposed. A set of problem instances of different sizes were generated and used for testing the algorithms. Experimental tests show that the revised ABC can significantly improve the performance of the algorithm in solving the capacitated vehicle routing problem. The ABC algorithm is improved in terms of better solutions achieved, greater robustness, and higher computational efficiency.

Acknowledgements

This project was supported by The Hong Kong Polytechnic University (Project No.: G-YJ35) and the Fundamental Research Funds for the Central Universities (Grant No. 2011ZM0038) in China.

References

[1] C. Malandraki, M. S. Daskin. Time dependent vehicle routing problems: formulations, properties and heuristic algorithms. Transportation Science, 1992, 26(3): 185-200.
[2] J. F. Cordeau, M. Gendreau, G. Laporte, J. Y. Rotvin, F. Semet. A guide to vehicle routing heuristics. Journal of the Operational Research Society, 2002, 53(5): 512-522.
[3] Arthur V. Hill, W. C. Benton. Modeling intra-city time-dependent travel speeds for vehicle scheduling problems. Journal of Operational Research Society, 1992, 43(4), 343-351.
[4] Soumia Ichoua, Michel Gendreau, Jean-Yves Potvin. Vehicle dispatching with time-dependent travel times. European Journal of Operational Research, 2003, 144(2): 379–396.
[5] Hideki Hashimoto, M. Yagiura, T. Ibaraki. An iterated local search algorithm for the time-dependent vehicle routing problem with time windows. Discrete Optimization, 2008, 5(2): 434-456.
[6] B. Fleischmann, M. Gietz, S. Gnutzmann, Time-dependent travel times in vehicle routing. Transportation Science, 2004, 38(2): 160-173.
[7] Dervis Karaboga. An idea based on honey bee swarm for numerical optimization. Technical Report TR06, Computer Engineering Department, Erciyes University, Turkey, 2005.
[8] Dervis Karaboga, Bahriye Basturk. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. Journal of Global Optimization, 2007, 39(3): 459–471.
[9] Alok Singh. An artificial bee colony algorithm for the leaf-constrained minimum spanning tree problem. Applied Soft Computing, 2009, 9(2): 625-631.