Chapter 9
Pheromone-Balance Driven Ant Colony Optimization with Greedy Mechanism

Masaya Yoshikawa

Abstract Ant colony optimization (ACO), which has been based on the feeding behavior of ants, has a powerful solution searching ability. However, since processing must be repeated many times, the computation process also requires a very long time. In this chapter, we discuss a new ACO algorithm that incorporates adaptive greedy mechanism to shorten the processing time. The proposed algorithm switches two selection techniques adaptively according to generation. In addition, the new pheromone update rules are introduced in order to control the balance of the intensification and diversification. Experiments using benchmark data prove the validity of the proposed algorithm.

1 Introduction

Combinatorial optimization problems can be used for a variety of fields, and various solutions for these types of problems have been studied. Among these solutions, a heuristic solution, referred to as meta-heuristics, has attracted much attention in recent years. Meta-heuristics is a general term for an algorithm that is obtained by technological modeling of biological behaviors [1–10] and physical phenomena [11]. Using this approach, an excellent solution can be obtained within a relatively short period of time. One form of meta-heuristics is ant colony optimization (ACO), which has been based on the feeding behavior of ants. The fundamental elements of ACO are the secretion of pheromone by the ants and its evaporation.

M. Yoshikawa
Department of information engineering, Meijo university, 1-501 Shiogamaguchi, Tenpaku Nagoya 468-8502, Japan
e-mail: evolution_algorithm@yahoo.co.jp

S.-I. Ao et al. (eds.), *Machine Learning and Systems Engineering*,
Lecture Notes in Electrical Engineering 68,
DOI 10.1007/978-90-481-9419-3_9,

Using these two elements, ants can search out the shortest route from their nest to a feeding spot. Fig. 1 shows the actual method for searching the shortest route.

Figure 1a shows the case where two ants A and B have returned from a feeding spot to the nest via different routes. In this case, ants A and B have secreted pheromone on their routes from the nest to the feeding spot and from the feeding spot to the nest. At this point, a third ant C moves from the nest to the feeding spot, relying on the pheromone trail that remains on the route.

ACO is performed based on the following three preconditions: (1) the amount of pheromone secreted from all ants is the same, (2) the moving speed of all ants is the same, and (3) the secreted pheromone evaporates at the same rate. In the above case, since the moving distance from the nest to the feeding spot is longer for route A than for route B, a larger amount of pheromone will have evaporated along route A than along route B, as shown in Fig. 1b. Therefore, ant C will select route B because a larger amount of pheromone remains on this route. However, ant C does not go to the feeding spot by a route that is the same as route B; rather, ant C goes to the feeding spot by route C and then returns to the nest, as shown in Fig. 1c. Another ant D then goes to the feeding spot either by route B or by route C (on which a larger amount of pheromone remains). The fundamental optimization mechanism of ACO is to find a shorter route by repeating this process.

For this reason, ACO has a powerful solution searching ability. However, since processing must be repeated many times, the computation process also requires a very long time. In order to shorten the processing time, this study proposes a new ACO algorithm that incorporates adaptive greedy selection. The validity of the proposed algorithm is verified by performing evaluation experiments using benchmark data.

This chapter is organized as follows: Section 2 indicates the search mechanism of ACO, and describes related studies. Section 3 explains the proposed algorithm with modified pheromone update rules. Section 4 reports the results of computer simulations applied to the travelling salesman problem (TSP) benchmark data. We summarize and conclude this study in Section 5.

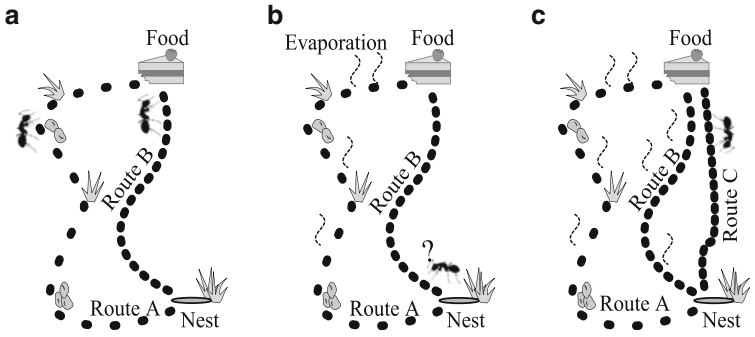

Fig. 1 Example of pheromone communication: (**a**) two routes on initial state, (**b**) positive feedback reinforcement using pheromone information, and (**c**) example of another route

2 Preliminaries

2.1 Ant Colony Optimization

ACO is a general term for the algorithm that can be obtained by technological modeling of the feeding behavior of ants. The basic model of ACO is the ant system (AS) [1] designed by M. Dorigo. The ant colony system (ACS) [2] is one of the modified algorithms of the AS. The ACS provided a better solution when the TSP was to be solved, compared to a genetic algorithm (GA) [9, 10] and a simulated annealing (SA) [11] method. Therefore, this study adopts ACS as its basic algorithm. Henceforth, ACO also denotes the ACS in this paper.

The processing procedure of ACO is explained using an example in which ACO is applied to the TSP. In ACO, each of several ants independently creates a travelling route (visiting every city just one time). At this time, each ant determines the next destination according to the probability p^k calculated from formula (1).

$$p^k(i,j) = \frac{[\tau(i,j)][\eta(i,j)]^\beta}{\sum_{l \in n^k} [\tau(l,j)][\eta(l,j)]^\beta} \tag{1}$$

Where, the value η (i,j) in formula (1) is called the static evaluation value. It is the reciprocal of the distance between city i and city j, and is a fixed value. On the other hand, τ (i,j) is referred to as the dynamic evaluation value. It expresses the amount of pheromone on the route between city i and city j, and this changes during the process of optimization. The term β represents a parameter and n^k represents a set of unvisited cities.

In ACO, the probability is high for selecting a route whose distance is short and whose pheromone amount is large. That is, selection using a roulette wheel is performed, as shown in Fig. 2.

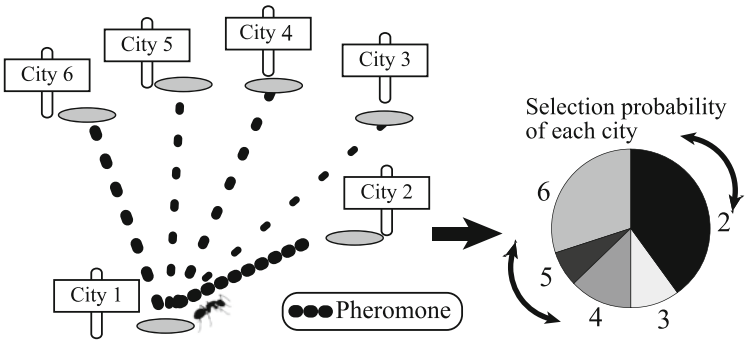

Fig. 2 Example of the selection probability

Here, the amount of pheromone on each partial route is determined by two types of rules: the local update rule and the global update rule. The local update rule is expressed by formula (2), and it is applied whenever each ant moves.

$$\tau(i,j) \leftarrow (1-\psi)\tau(i,j) + \psi\tau_0 \tag{2}$$

In formula (2), ψ is a decay parameter in local update rule, τ_0 is the initial value of pheromone. The global update rule is expressed by formula (3), and it is applied after each ant completes a travelling route.

$$\tau(i,j) \leftarrow (1-\rho)\tau(i,j) + \rho\Delta\tau(i,j) \tag{3}$$

$$\Delta\tau(i,j) = \begin{cases} 1/L^+ & \text{if}(i,j) \in T^+ \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

In formula (3), ρ represents the evaporation rate of pheromone in the global update rule, T^+ represents the shortest travelling route among the completed travelling routes, and L^+ represents the length of the partial route that composes the shortest travelling route.

In ACO, the amount of pheromone is larger on a route that has been more often selected, and is smaller, due to evaporation, on a route that has been less often selected. In addition, a rule called pseudo-random-proportional rule is also introduced. For this rule, an ant is forced to move to a city, with the selection probability being the highest at a certain probability. In other words, a random number q is generated between 0 and 1. If q is smaller than threshold value q_t, an ant will move to a city whose value of numerator in formula (1) is the largest. If q is larger than the threshold value, the usual selection by formula (1) will be performed.

2.2 Related Studies

Many studies [1–8] have been reported for the use of ACO, such as those that have applied ACO to network routing problems [3] and to scheduling problems [4]. A number of studies [5, 6] have also been reported on pheromone, which is an important factor for controlling the intensification and diversification of a search. Hybridization with other algorithms, such as a GA [7] or SA [8] method, has also been studied. However, no study has yet been reported that incorporates adaptive greedy selection into an ACO, as is proposed in this study.

3 Hybrid ACO with Modified Pheromone Update Rules

In ACO, when x represents the number of cities, y represents the total number of ants, and z represents the number of processing repetitions, the number of calculations for formulae (1), (2), and (3) are expressed by $(x - 1) \times y \times z$, a $x \times y \times z$, and z, respectively. Since the number of processing repetitions in ACO is large, a processing time problem is inherent in ACO. In this study, incorporating a greedy selection mechanism into the ACO reduced the number of calculations in formulae (1) and (2). Consequently, the processing time was shortened.

A greedy selection approach only employs static evaluation values, with no use of dynamic evaluation values, when selecting the next destination city. That is, an ant moves from the present city to the nearest city. Since static evaluation values are constant in the optimization process, once the calculation is performed in the early stage, it is not necessary to re-calculate. Therefore, the processing time can be shortened by performing a greedy selection. However, since greedy selection favors the local optimal solution, the balance between formula (1) and the greedy selection becomes important.

In order to control this trade-off relationship, the greedy selection is adaptively used in this study. The adaptive greedy selection is explained using the TSP of six cities, as shown in Fig. 3.

In Fig. 3, when city A is set as the start city, the following three selections are performed using formula (1): (1) movement from city A to city B; (2) movement from city B to city C; and (3) movement from city C to city D. The greedy selection is also applied to the other movements; namely, from city D to city E and from city E to city F.

In contrast, when city D is set as the start city, the following three selections are performed using the greedy selection: (1) movement from city D to city E,

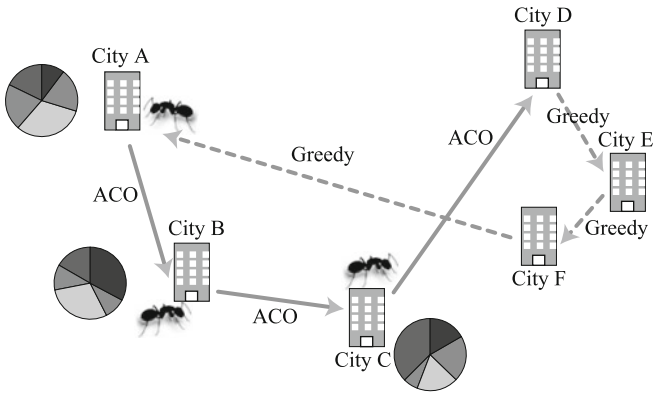

Fig. 3 Example of adaptive greedy selection

(2) movement from city E to city F, and (3) movement from city F to city A. Formula (1) is applied to the other movements; namely, from city A to city B and from city B to city C. In this study, the proportion of the greedy selection and the selection by formula (1) is changed according to generation, as shown in Fig. 4.

Figure 4 shows an example obtained when the total number of processing repetitions was set at 1,000, and the proportion of the greedy selection and the selection by formula (1) was changed every 200 repetitions. In this example, the selection by formula (1) was performed until the 200th repetition. From the 201st repetition to the 400th repetition, the selection by formula (1) was applied to 80% of cities, and the greedy selection was applied to 20% of cities. In other words, for the problem of 100 cities, the selection by formula (1) was performed on 80 cities and the greedy selection was performed on 20 cities. Similarly, from the 401st repetition to the 600th repetition, the selection by formula (1) was applied to 60% of cities, and the greedy selection was applied to 40% of cities. Thus, by changing the proportion of two selection methods according to generation, the trade-off relationship between the accuracy of the solution and the processing time can be controlled.

In optimization algorithms, controlling the trade-off relationship between the intensification and diversification of search is also important. In the adaptive greedy selection proposed in this study, the pressure of intensification is strong. Therefore, in order to strengthen the pressure of diversification, the local update rule and the global update rule are partially changed.

First, the local update rule is generally applied whenever an ant moves. However, it is not applied when the greedy selection is performed. Next, the global update rule is generally applied to the best travelling route; i.e., the shortest travelling route. It is also applied to the second shortest travelling route. By adding these two modifications, the proposed algorithm can realize a balance between the intensification and diversification of the search.

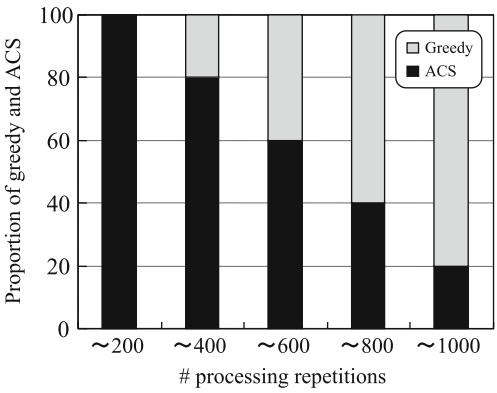

Fig. 4 Example of the proportion of the greedy selection and the selection by formula (1)

4 Experiments and Discussion

In order to verify the validity of the proposed algorithm, several comparison experiments were performed. Each experiment was performed 10 times. Table 1 shows the conditions used in the experiments. In the table, the designation ACS/Greedy indicates that the city selection of the first half was performed using formula (1), and that of the second half was performed using the greedy selection. Similarly, the designation 80/20 indicates that 80% of the cities were selected using formula (1), and selection of 20% of the cities used the greedy selection. In each table, technique (a) represents the conventional ACO. Techniques (b) and (c) represent algorithms in which the proportion of the greedy selection and the selection by formula (1) was not changed according to generation. Techniques (d), (e), (f), and (g) represent algorithms in which the proportion was changed according to generation. These four algorithms are proposed in this study. Tables 2–4 show the experimental results.

As shown in Tables 2–4, the processing time was shortened when the greedy selection was used, regardless of its proportion. By changing the proportion of the greedy selection and the selection by formula (1) according to generation, the ability to search a solution also improved, when compared with the case where the proportion was fixed. In order to examine the optimal proportion, changes in solutions obtained using techniques (d), (e), (f), and (g) were plotted, as shown in Fig. 5. In this figure, the horizontal axis represents the number of processing

Table 1 Experimental conditions

Technique	Generation (# repetitions)				
	1–200	201–400	401–600	601–800	801–1000
(a) ACS			(N/A)		
(b) Greedy/ACS			50/50		
(c) ACS/Greedy			50/50		
(d) Greedy/ACS	0/100	20/80	40/60	60/40	80/20
(e) ACS/Greedy	100/0	80/20	60/40	40/60	20/80
(f) Greedy/ACS	80/20	60/40	40/60	20/80	0/100
(g) ACS/Greedy	20/80	40/60	60/40	80/20	100/0

Table 2 Results of $q_t = 0.25$

Technique	Best	Average	Worst	Time (s)
(a) ACS	21839	22049	22393	20.00
(b) Greedy/ACS	23196	23482	23690	14.73
(c) ACS/Greedy	22348	22599	22915	15.39
(d) Greedy/ACS	22103	22295	22619	15.64
(e) ACS/Greedy	21997	22264	22709	16.35
(f) Greedy/ACS	21967	22161	22535	15.93
(g) ACS/Greedy	21731	22109	22543	16.42

Table 3 Results of $q_t = 0.50$

Technique	Best	Average	Worst	Time (s)
(a) ACS	21700	21876	22056	18.90
(b) Greedy/ACS	23082	23490	23700	14.24
(c) ACS/Greedy	22348	22707	23122	14.85
(d) Greedy/ACS	21665	22114	22330	15.36
(e) ACS/Greedy	21866	22062	22386	15.72
(f) Greedy/ACS	21824	22026	22346	15.31
(g) ACS/Greedy	21817	22149	22721	15.81

Table 4 Results of $q_t = 0.75$

Technique	Best	Average	Worst	Time (s)
(a) ACS	21558	21768	22262	18.14
(b) Greedy/ACS	23218	23561	23773	13.80
(c) ACS/Greedy	22336	22978	23429	14.37
(d) Greedy/ACS	21787	22038	22503	14.78
(e) ACS/Greedy	21713	21888	22181	15.21
(f) Greedy/ACS	21826	21901	21977	14.73
(g) ACS/Greedy	21665	21975	22312	15.27

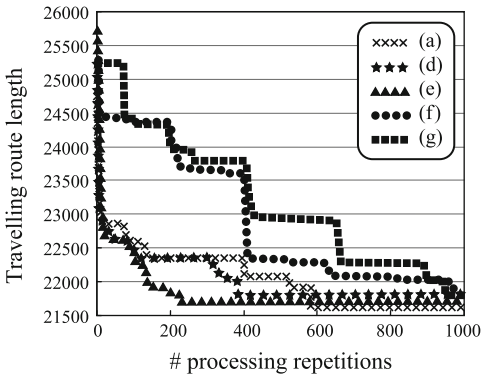

Fig. 5 Comparisons of selective techniques (d), (e), (f), and (g)

repetitions and the vertical axis represents the travelling route length of the optimal solution.

As shown in Fig. 5, when the number of processing repetitions was small as the termination condition, techniques (d) and (e) showed excellent performance. When the number of processing repetitions was sufficient, techniques (f) and (g) also exhibited excellent performance. Thus, the selection of technique according to the given experimental condition (termination condition) was clearly important.

5 Conclusion

In this chapter, we proposed a new technique that incorporates a greedy mechanism to shorten the processing time of an ACO. A hybrid technique that uses two types of selection techniques (i.e., the conventional selection technique and the greedy selection technique) is introduced to generate one solution. A technique was also developed for adaptively changing these two selection techniques according to generation. In order to control the balance of the intensification and diversification, the update rule of pheromone was improved in each phase of global update and local update. The validity of the proposed technique was verified by comparative experiments using benchmark data. As a future task, the proposed technique should be applied to practical problems other than the TSP.

References

1. M. Dorigo, V. Maniezzo, A. Colorni, Ant system: optimization by a colony of cooperating agents. IEEE Trans. Syst. Man Cybern. Part B. **26**(1), 29–41 (1996)
2. M. Dorigo, L.M. Gambardella, Ant colony system: a cooperative learning approach to the traveling salesman problem. IEEE Trans. Evol. Comput. **1**(1), 53–66 (1997)
3. T.H. Ahmed, Simulation of mobility and routing in ad hoc networks using ant colony algorithms. Proc. Int. Conf. Inf. Technol. Coding Comput. **2**, 698–703 (2005)
4. M. Yoshikawa, H. Terai, A Hybrid Ant Colony Optimization Technique for Job-Shop Scheduling Problems, Proc. of 4th IEEE/ACIS International Conference on Software Engineering Research, Management & Applications (SERA 2006), 95–100 (2006)
5. A. Hara, T. Ichimura, N. Fujita, T. Takahama, "Effective Diversification of Ant-Based Search Using Colony Fission and Extinction", Proc, of IEEE Congress on Evolutionary Computation (CEC 2006), 1028–1035 (2006)
6. Z. Ren, Z. Feng, Z. Zhang, GSP-ANT: an efficient ant colony optimization algorithm with multiple good solutions for pheromone update. Proc. IEEE Int. Conf. Intelligent Comput. Intelligent Syst. **1**, 589–592 (2009)
7. G. Shang, J. Xinzi, T. Kezong, "Hybrid Algorithm Combining Ant Colony Optimization Algorithm with Genetic Algorithm", Proc. of Chinese Control Conference (CCC 2007), 701–704 (2007)
8. Y.-H. Wang, P.-Z. Pan, A Novel Bi-Directional Convergence Ant Colony Optimization with SA for Job-Shop Scheduling, Proc. of International Conference on Computational Intelligence and Software Engineering (CiSE 2009), 1–4 (2009)
9. J. Holland, *Adaptation in Natural Artificial Systems*, 2nd edn. (The University of Michigan Press, MIT Press, 1992)
10. D.E. Goldberg, *Genetic Algorithms in Search Optimization, and Machine Learning* (Addison Wesley, Reading MA, 1989)
11. R.A. Rutenbar, Simulated annealing algorithms: an overview. IEEE Circuits Dev. Mag. **5**(1), 19–26 (1989)