CHAPTER 5

Fundamentals of Ant Colony Search Algorithms

YONG-HUA SONG, HAIYAN LU, KWANG Y. LEE, and I.K. YU

5.1 INTRODUCTION

The ant colony optimization (ACO) is a meta-heuristic approach for combinatorial optimization problems, which can be regarded as a paradigm for all ant colony search algorithms that are inspired by the foraging behavior of the social insects, especially the ants [1]. From a broader perspective, the ACO algorithms belong to the class of model-based search (MBS) algorithm according to Dorigo [2]. MBS algorithms have become increasingly popular methods for solving combinatorial optimization problems. An MBS algorithm is characterized by the use of a (parameterized) probabilistic model that is used to generate solutions to the problem under consideration. The MBS algorithms can be classified into two categories based on how the probabilistic model is used: (i) the algorithms use a given probabilistic model without changing the model structure during run-time; and (ii) the algorithms use and change the probabilistic model in alternating phases. The ACO algorithms fall into the first category. At run-time, ACO algorithms will update the parameters' values of the probabilistic model in such a way that there will be more chances to generate high-quality solutions over time.

Ant colony search algorithms (ACSAs) have recently been introduced as powerful tools to solve a diverse set of optimization problems, such as the traveling salesman problem (TSP) [3, 4], the quadratic assignment problem [4, 5], and optimization problems in power systems [6–15, 23, 24], such as the generation scheduling problem, unit commitment problem (economic dispatch problem) [16–18], and web usage mining problem [19]. This chapter focuses on the fundamentals of the ACSAs. The applications of ACSAs in power systems will be covered in depth in Chapter 20.

Modern Heuristic Optimization Techniques. Edited by K. Y. Lee and M. A. El-Sharkawi

5.2 ANT COLONY SEARCH ALGORITHM

The first ACS algorithm was proposed by Dorigo in the early 1990s. The ACS belongs to biologically inspired heuristic algorithms. It was developed mainly based on the observation of the foraging behavior of a real ant. It will be useful to understand how ants, which are almost blind animals with very simple individual capacities acting together in a colony, can find the shortest route between the ant nest and a source of food. Section 5.2.1 describes the behavior of real ants, followed by the presentation of the two most (experimentally) successful ant colony search algorithms, with their progenitor in Section 5.2.2. The major characteristics of the ACSAs are summarized in the Section 5.2.3.

5.2.1 Behavior of Real Ants

As is well-known, real ants are capable of finding the shortest path from food sources to the nest without using visual cues. They are also capable of adapting to changes in the environment, for example, finding a new shortest path once the old one is no longer feasible due to a new obstacle. Figure 5.1 clearly illustrates these phenomena. In Fig. 5.1a, ants are moving on a straight line that connects a food source (A) to the nest (E). Once an obstacle appears as shown in Fig. 5.1b, the path is cut off. Shortly, the ants can establish a new shortest path as shown in Fig. 5.1c.

The studies by ethnologists reveal that such capabilities that ants have are essentially due to what is called *pheromone trails*, which ants use to communicate information among individuals regarding the walking path or the decision about where to go when they are foraging. According to Blum and Dorigo [2], real ants initially explore the area surrounding their nest in a random manner when searching for

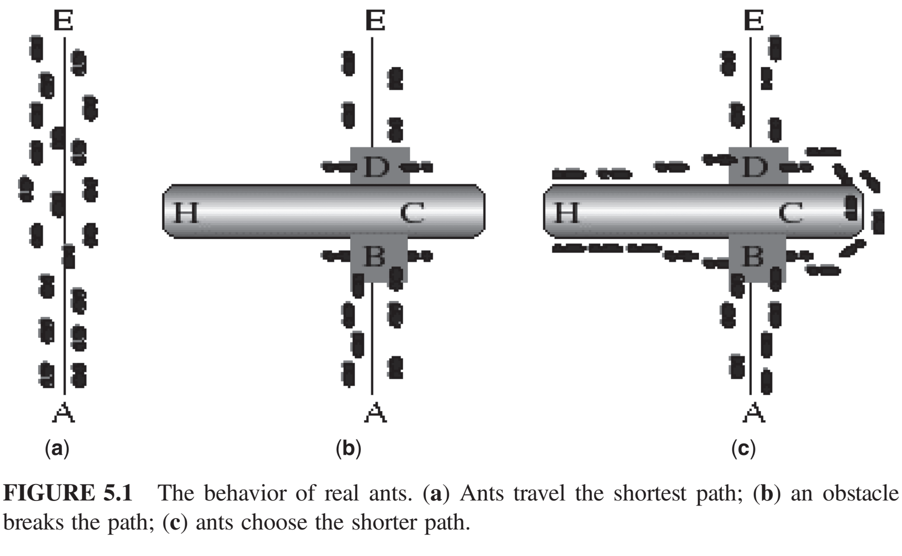

FIGURE 5.1 The behavior of real ants. (**a**) Ants travel the shortest path; (**b**) an obstacle breaks the path; (**c**) ants choose the shorter path.

food. As soon as an ant finds a food source, it carries some of the found food to the nest. While it is walking, the ant deposits a chemical pheromone trail on the ground. The pheromone trails deposited on the ground will guide other ants to the food source. Each ant probabilistically prefers to follow a direction rich in pheromone rather than a poorer one. The indirect communication between the ants via the pheromone trails allows them to find the shortest paths between their nest and food sources.

The process illustrated in Fig. 5.1 can then be explained as follows: In Fig. 5.1a, the path between the food source and the nest is clear. The ants carrying the food walk back to the nest in a random fashion initially. Supposing that all ants walk at approximately the same speed, the shortest path will have most ants visited on average. Therefore, the pheromone on the shortest path will accumulate faster than on any other paths. After a short transitory period, the difference in the amount of pheromone on the paths is sufficiently noticeable to the new ants coming into the system. From that point of time, new ants will preferably walk along the shortest path, which is a straight line that connects a food source to the nest. When an obstacle blocks an established path, as shown in Fig. 5.1b, the pheromone trail is interrupted. Those ants that are just in front of the obstacle cannot continue to follow the pheromone trail and therefore they would choose to turn right or turn left randomly as they do not have any clue about which is the best choice. In Fig. 5.1c, those ants that choose by chance the shorter path around the obstacle will more rapidly establish the interrupted pheromone trail compared with those that choose the longer path as there are more ants walking along the shorter path at each time unit. Hence, the shorter path will have a higher amount of pheromone deposited on average, and this will in turn cause a higher number of ants to choose the shorter path. Due to this positive feedback (autocatalytic) process, very soon all ants will choose the shorter path.

5.2.2 Ant Colony Algorithms

The foraging behavior of real ants has inspired the ant colony algorithms, which are algorithms in which a set of artificial ants cooperate to find the solution to a combinatorial optimization problem by changing information via pheromone deposited on the artificial paths.

For the ease of description of the ant colony algorithms, we look at a discrete optimization problem $\wp = (S, f, \Omega)$ characterized [1] as

- A finite set $C = \{c_1, c_2, \ldots, c_{Nc}\}$ of solution components.
- A finite set $L = \{l_{i,j} | (c_i, c_j) \in \tilde{C}\}$ of possible connections among the elements of $\tilde{C}$, which is a subset of $C \times C$ (the Cartesian product of C and C), where $l_{i,j}$ indicates the connection between components c_i and c_j. The number of possible connections is denoted by $|L| \leq N_C^2$.
- A finite set $\Omega \equiv \Omega(C, L)$ of constraints assigned over the elements of C and L.
- A finite set χ of states of the problem, defined in terms of all possible sequences $s = \langle c_i, c_j, \ldots, c_k, \ldots \rangle$ over the elements of C. The number of components in the

sequence is denoted by $|s|$. The maximum length of a sequence is bounded by a positive constant number $N < +\infty$.

- A set $\tilde{\chi}$ of all states that are feasible with respect to the constraints $\Omega(C, L)$, $\tilde{\chi} \subseteq \chi$.
- A set $S = \{s|s \in \tilde{\chi}\}$ of (candidate) solutions, $S \subseteq \chi$.
- $J_{i,j}$ is a connection cost associated with each $l_{i,j} \in L$.
- $J_s(L)$ is a cost associated with the solution s, which can be calculated from all the costs $J_{i,j}$ of all the components belonging to the solution s.
- A neighborhood structure of components: Given two components, c_1 and c_2, the component c_2 is the component c_1 if both components are in C, and the component c_2 can be reached from the component c_1 in one logical step. The set of all neighbor components of the component c is denoted by N_c.
- Each connection $l_{i,j}$ is weighted by $\tau_{i,j}$.

In order to solve the above discrete problem, it is often useful to map the problem on a graph $G = (C, L)$ and refer to this graph as the construction graph. The solutions to the optimization problem can be expressed in terms of feasible paths on the graph G, and the optimal solution will be the path that yields minimum connection cost.

There are a number of experimental successful ant algorithms published in the literature that can be used to solve the above problem [20]. Here we describe two of the most successful ant algorithms, namely, the ant colony system (ACS) and the max-min ant system (MMAS), with their progenitor, the ant system (AS) for the sake of application of ant colony search algorithms in solving combinatorial optimization problems facing power systems.

These three algorithms share the same algorithmic skeleton as shown in Fig. 5.2. The skeleton consists of three major phases, which are the initialization phase, the solution construction phase, and the pheromone updating phase and other optional phases, such as a phase that locally updates the pheromone trails, as shown in the dashed-line block in the figure. The initialization phase does almost the same tasks as listed below across all three algorithms:

- Set up a construction graph, which is a completely connected and weighted graph $\varsigma = (C, L, T)$, where the vertices are the components C, the set L fully connects the components C, and T is a vector whose components representing the so-called pheromone trail strength, τ, which are considered to be associated with possible connections (or edges) only. This task is often the most difficult task when applying an ant colony search algorithm to an optimization problem.
- Initialize the pheromone trail strength for all the edges. Except for the MMAS in which the initial pheromone trail strength for all the edges is set to be a large value, the AS and ACS initialize these values to be a small value, in general.
- Set the number of artificial ants (ants for abbreviation) in a colony as m, and put each ant on a randomly chosen vertex.

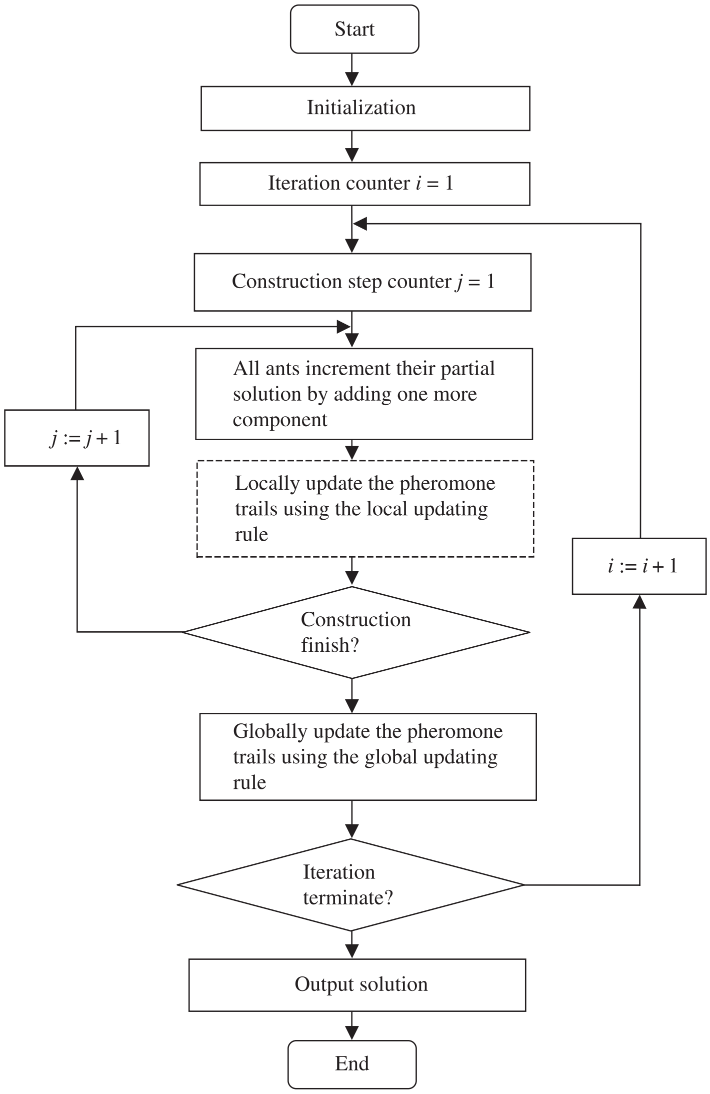

FIGURE 5.2 Structure of simple ACSA.

- Set up the termination criteria for the iteration looping, which may be that the iteration number exceeds a predefined maximum number of solution construction steps or that the computation time has exceeded a given CPU-time limit.

When the solution construction phase starts, m ants have been put on randomly chosen vertices on the construction graph, and their paths consist of their initial vertices. Within each construction step, all the ants construct their feasible paths (partial solutions) by moving to the next vertex based on a probabilistic decision according to the so-called state transition rule. The state transition rules are different among various ant search algorithms. However, they all serve the purpose of guiding the ants to make a more cost-effective move from their current positions. After all the ants have made one move, their current feasible paths may be improved by applying local pheromone updating rule depending on the ant search algorithm under consideration. The solution construction phase is repeated until all ants have completed their feasible paths. Then the pheromone trails will be globally updated using the global pheromone updating rule. The global updating rule enforces two things: one is the pheromone evaporation, which stops pheromone trails from unlimited accumulation, and the other is the pheromone reinforcement, which makes the favorite edges have stronger pheromone trails. The global updating rule can be different from one algorithm to another, but they all aim to fade out the pheromone trails on the edges that are not used by ants exponentially and to enforce the favorite edges so that they can attract more ants in following construction steps.

In the rest of this subsection, we will describe how these three algorithms work with emphasis on the state transition rules and the pheromone updating rules they are applying.

5.2.2.1 The Ant System The ant system (AS) was the first ant colony search algorithm proposed as a novel heuristic method for solving combinatorial problems in the early 1990s [1, 21, 22]. Informally, an AS can be described as follows: After the initialization phase, m ants are put on their initial vertex on the construction graph, and all pheromone trails are initialized to be a small value (i.e., within the range from 0 to 1) for all the edges. In each solution construction step, each ant incrementally constructs a solution by adding one solution component to the partial solutions it has constructed so far respectively. Suppose that the kth ant is at the vertex c_i in the construction step t. The kth ant performs a randomized walk from the vertex c_i to the next vertex to construct a feasible solution incrementally in such a way that the next vertex is chosen stochastically according to the transition probabilities of candidate vertices with respect to the current vertex of the kth ant. The probability with which the kth ant at the vertex i chooses to move to the vertex j can be determined by the random-proportional state transition rule [3] as shown below:

$$p_k(i,j) = \begin{cases} \dfrac{[\tau(i,j)]^{\alpha} \cdot [\eta(i,j)]^{\beta}}{\sum_1^t [\tau(i,u)]^{\alpha} \cdot [\eta(i,u)]^{\beta}} & j, u \in N_{k,i} \\ 0 & \text{otherwise,} \end{cases} \tag{5.1}$$

where $\tau(i,j)$ represents the pheromone trail associated with $l_{i,j}$, which is the connection between vertices i and j, $\eta(i,j)$ a heuristic value, called the desirability of adding connection $l_{i,j}$ to the solution under construction and can be determined according to the optimization problem under consideration. It is usually set to be the reverse of the connection cost, $J_{i,j}$, associated with the edge $l_{i,j}$, that is, $1/J_{i,j}$, $N_{k,i}$ is the feasible neighbor components of the kth ant at the vertex c_i with respect to the problem constraints Ω, $N_{k,i} \subseteq N_i$, α and β are two parameters that determine the relative importance of pheromone versus the heuristic value ($\alpha, \beta > 0$).

Once all ants have completed their solutions, the pheromone trails are updated on all edges according to the pheromone global updating rule as shown below:

$$\tau(i, j) \leftarrow (1 - \alpha) \cdot \tau(i, j) + \sum\nolimits_{k=1}^{m} \Delta\tau_k(i, j), \tag{5.2}$$

where α is a pheromone decay parameter with $0 < \alpha \leq 1$, and

$$\Delta\tau_k(i,j)\begin{cases} 1/J_{i,j} & (i,j) \in \varphi \\ 0 & \text{otherwise}, \end{cases} \tag{5.3}$$

with φ being the set of moves done by the kth ant.

The introduction of the heuristic value $\eta(i, j)$ in (5.1) is mainly to favor the edges that are cost-effective and also have a greater amount of pheromone. The first term in (5.2) models the pheromone evaporation, in which the pheromone trails are lowered by the pheromone decay factor. The pheromone trails for the unfavorite edges will be evaporated exponentially. The second term in (5.2) models the reinforcement of pheromone trails. It makes the edges that have been visited by ants become more desirable for future ants. Ants deposit an amount of pheromone related to the quality of the solutions that they produced so that the more cost-effective solution generated by an ant, the greater the amount of pheromone it deposits on the edges that it used to produce the solution. According to Dorigo and Gambardella [3], the pheromone updating rule was meant to simulate the change in the amount of pheromone due to both the addition of new pheromone deposited by ants on the visited edges and the pheromone evaporation.

Although the AS performance was not competitive with the state-of-the-art algorithms for some benchmark problems, such as the traveling salesman problem (TSP), it gave rise to a whole set of successful ant colony search algorithms that have recently been unified in a novel meta-heuristic called ant colony optimization (ACO).

5.2.2.2 The Ant Colony System The ant colony system (ACS) was developed after the AS. The ACS can outperform the AS due to the following three modifications [3]: (i) the state transition rule provides a direct way to balance exploration of new edges and exploitation of a found solution and accumulated knowledge about the problem; (ii) the global pheromone updating rule is applied only to the edges that belong to the best journey; and (iii) while ants construct a solution, a local pheromone updating rule is applied for each construction step.

Informally, the ACS algorithm can be described as follows: m ants are initially put on randomly chosen vertices on the construction graph. Each connection or edge on the graph is associated with a pheromone trail that is initialized to be a small positive value. Each ant walks randomly on the graph to construct its solution by repeatedly adding one solution component to its partial solution following the state transition rule, which is different from the one used in AS. Suppose that the kth ant is at the vertex c_i and the construction has been up to the tth step. The kth ant will move to the next vertex to increment its solution by one component. Its next vertex, say c_j, can be determined by

$$c_j = \begin{cases} \arg\max_{u \in N_{k,i}} \{[\tau(i,u)]^{a} \cdot [\eta(i,u)]^{\beta}\} & q \leq q_0 \\ c^* & \text{otherwise,} \end{cases} \tag{5.4}$$

where q is a uniformly distributed random number within [0, 1], q_0 is a parameter $(0 < \alpha < 1)$, which determines the relative importance of local exploitation versus global exploration, and c^* is a random variable selected according to the probability distribution given in (5.1). The state transition rule resulting from (5.1) and (5.4) is called *pseudo-random-proportional rule* [3]. This transition rule favors moves to vertices connected by more cost-effective edges and with denser pheromone trails. By applying the *pseudo-random-proportional rule*, for each construction step, an ant at the vertex c_i has to choose a vertex c_j to move to; at that moment, a uniformly distributed random number $q_0 \in [0, 1]$ is generated. If $q \leq q_0$, then the next vertex will be the vertex determined by (5.4), which yields the best move, otherwise, it will be the component c^* chosen according to (5.1). While ants construct their solutions, for each step, the pheromone trails for the edges the ants have visited will be changed by applying a locally update rule shown below:

$$\tau(i,j) \leftarrow (1-\rho) \cdot \tau(i,j) + \rho \cdot \Delta\tau(i,j), \tag{5.5}$$

where ρ with $0 < \rho < 1$, is a parameter, and $\Delta\tau(i, j)$ is the amount of pheromone that kth ant puts on the edge $l_{i,j}$. The pheromone update $\Delta\tau(i,j)$ can be set to be the value of the initial pheromone trails for the simple ACS algorithm [3].

Once all the ants have completed their solutions, the pheromone trails will be updated by the pheromone global updating rule shown below:

$$\tau(i,j) \leftarrow (1-\alpha) \cdot \tau(i,j) + \alpha \cdot \Delta\tau_k(i,j), \tag{5.6}$$

where α is a pheromone decay parameter with $0 < \alpha \leq 1$, and

$$\Delta\tau_k(i,j) = \begin{cases} (J_{gb})^{-1} & l_{i,j} \in s^* \\ 0 & \text{otherwise,} \end{cases} \tag{5.7}$$

where J_{gb} is the cost of the globally best ant solution, s^*, from the beginning of the trial.

5.2.2.3 *The Max-Min Ant System* According to Stützle and Hoos [4], the max-min ant system (MMAS) is an ACO algorithm derived from AS and is one of the best ACO algorithms for the TSP and the quadratic assignment problem (QAP), which is a typical benchmark problem for challenging any combinatorial optimization algorithms.

The MMAS differs from the AS in following three key aspects [4]:

- To exploit the best solution found during an iteration or during the run of the algorithm, after each iteration only one single ant adds pheromone. This ant may be the one that found the best solution in the current iteration (iteration-best ant) or the one that found the best solution among all the iterations from the beginning of the trial (global-best ant), which is the same one used in the state transition rule of ACS.
- To avoid stagnation of the search, the range of possible pheromone trails (edges in the construction graph) on each solution component is limited to an interval $[\tau_{\min}, \tau_{\max}]$.
- Initialize the pheromone intensity for all edges to be $\tau_{\max}$ to achieve a higher exploration of solution at the beginning of trails.

Informally, the MMAS algorithm can be described as follows: m ants are initially positioned at randomly chosen vertices on the construction graph. The pheromone trails for all the connections (or edges) are initialized to be an arbitrarily large value such that all pheromone trails are the upper limit of all possible pheromone trail values. Each ant walks randomly on the graph to construct its solution by repeatedly adding one solution component to its current partial solution following the state transition rule, which is the same as the one used in the AS. Suppose that the kth ant is at the vertex c_i and the construction has been up to the tth step. The kth ant will move to the next vertex to increment its partial solution by one component. The next vertex is chosen stochastically according to the transition probabilities of candidate vertices with respect to the current vertex of the kth ant. The probability with which the kth ant at the vertex i chooses to move to the vertex j can be determined by the random-proportional state transition rule [3] as shown in (5.1).

Once all ants complete their solutions, in other words, the program completes one iteration, the pheromone trails consisting of the best solution will be updated by applying a global update rule shown below:

$$\tau(i, j) \leftarrow (1 - \alpha) \cdot \tau(i, j) + \Delta\tau^{\text{best}}(i, j), \tag{5.8}$$

where α with $0 < \alpha \leq 1$ is a pheromone decay parameter, and $\Delta\tau^{\text{best}}(i, j) = 1/f(s^{\text{best}})$ and $f(s^{\text{best}})$ denotes the solution cost of either the iteration-best solution (s^{ib}) or the global-best solution (s^{gb}). The MMAS mainly uses the iteration-best solution for $f(s^{\text{best}})$ whereas the ACS uses only the global-best solution in global pheromone update.

After updating the pheromone trails, the pheromone trail constraint $\tau_{\min} \leq \tau(i, j) \leq \tau_{\max}$, where $\tau(i, j)$ represents the pheromone trails for the connection $l_{i,j}$, will be enforced such that if $\tau(i, j) < \tau_{\min}$, $\tau(i, j)$ will be re-set to be $\tau_{\min}$ and if $\tau(i, j) > \tau_{\max}$, $\tau(i, j)$ will be re-set to be $\tau_{\max}$.

The iterations continue until the termination criterion is met and one run completes.

5.2.3 Major Characteristics of Ant Colony Search Algorithms

There are some attractive properties of ant search algorithms when compared with other search methods:

5.2.3.1 Distributed Computation: Avoid Premature Convergence Conventionally, scientists choose to work on a system simplified to a minimum number of components in order to observe essential information. An ant search algorithm often simplifies as much as possible the components of the system, for the purpose of taking into account their large number. The power of the massive parallelism in ant algorithms is able to deal with incorrect, ambiguous, or distorted information, which is often found in nature. The computational model contains the dynamics that is determined by the nature of local interactions between many elements (artificial ants).

5.2.3.2 Positive Feedback: Rapid Discovery of Good Solution The ant search algorithms make use of a unique indirect communication means to share global information while solving a problem. Occasionally, the information exchanged may contain errors and should alter the behavior of the ants receiving it. As the search proceeds, the new population of ants often containing the states of higher fitness will affect the search behavior of others and will eventually gain control over other agents while at the same time actively exploiting inter-ant communication by mean of the pheromone trail laid on the path. The artificial ant foraging behavior dynamically reduces the prior uncertainty about the problem at hand. As ants doing a task can be either "successful" or "unsuccessful," they can switch between these two according to how well the task is performed. Unsuccessful ants also have a certain chance to switch to be inactive, and successful ants have a certain chance to recruit inactive ants to their task. Therefore, the emerging collective effect is in a form of autocatalytic behavior, in that the more ants following a particular path, the more attractive this path becomes for the next ants that meet. It can give rise to global behavior in the colony.

5.2.3.3 Use of Greedy Search and Constructive Heuristic Information: Find Acceptable Solutions in the Early Stage of the Process Based on the available information collected from the path (pheromone trail level and heuristic information), the decision is made at each step as a constructive way by the artificial ants, even if each ant's decision always remains probabilistic. It tends to evolve a group of initial poorly generated solutions to a set of acceptable solutions through successive generations. It uses objective function to guide the search only and does not need any other auxiliary knowledge. This greatly reduces the complexity of the problem. The user only has to define the objective function and the infeasible regions (or obstacle on the path).

5.3 CONCLUSIONS

This chapter focuses the fundamentals of ant colony search algorithms by presenting two major ACSAs, which are the ACS and the MMAS. The individual ants are rather simple; however, the entire colony foraging toward the bait site can exhibit complicated dynamics resulting in a very attractive search capability. The ACSAs consist of three major phases: (i) the initialization phase, in which the construction graph is built based on the combinatorial optimization problem at hand and related parameters are initialized; (ii) the solution construction phase, in which each ant constructs its own solution to the problem incrementally following the state transition rule until all ants complete their solutions; (iii) the pheromone update phase, in which the pheromone trails corresponding with each connection in the construction graph will be updated based on the global pheromone trail updating rule. The state transition rule and the pheromone trail updating rule can be different from one algorithm to the other. The ACSAs have attractive features compared with other heuristic search methods. The important features are (i) distributed computation in nature, (ii) autocatalytic process, and (iii) use of greedy search and constructive heuristic information. The experimental studies on the typical benchmark problems, such as the TSP and QAS, have shown that the ACSAs are promising in solving combinatorial optimization problems.

REFERENCES

1. Dorigo M, Di Caro G. Ant colony optimization: a new meta-heuristic. Proceedings of the 1999 Congress on Evolutionary Computation, 1999 (CEC'99), the IEEE; Vol. 2. 6–9 July 1999. p. 1470–1477.
2. Blum C, Dorigo M. The hyper-cube framework for ant colony optimization. IEEE Trans Systems Man Cybernetics B 2004; 34(2):1161–1172.
3. Dorigo M, Gambardella LM. Ant colony system: A cooperative learning approach to the traveling salesman problem. IEEE Trans Evol Computat 1997; 1(1):53–66.
4. Stützle T, Hoos HH. MAX–MIN ant system. Future Generation Computer Systems 2000; 16(8):889–914.
5. Maniezzo V, Colorni A. The ant system applied to the quadratic assignment problem. IEEE Trans Knowledge Data Eng 1999; 11(5):769–778.
6. Song YH, ed. Modern optimisation techniques in power systems. Boston: Kluwer Academic Publishers; 1999.
7. Irving MR, Song YH. Optimisation techniques for electrical power systems. IEE Power Eng J 2000; 14(5):245–254.
8. Song YH, Irving MR. Optimisation techniques for electrical power systems. Part 2: Heuristic optimisation techniques. IEE Power Eng J 2001; 15(3):151–160.
9. Song YH, Chou CS. Application of ant colony search algorithms in power system optimisation. IEEE Power Eng Rev 1998; 18(12):63–64.
10. Song YH, Chou CS, Min Y. Large-scale economic dispatch by artificial ant colony search algorithms. Electric Machines and Power Systems 1999; 27(5):679–690.

11. Yu IK, Song YH. A novel short-term generation scheduling technique of thermal units using ant colony search algorithms. Electric Power and Energy Systems 2001; 23:471–479.

12. Song YH, Chou CS, Stonham TJ. Combined heat and power economic dispatch by improved ant colony search algorithm. Electric Power Systems Research 1999; 52:115–121.

13. Yu I-K, Chou CS, Song YH. Application of the ant colony search algorithm to short-term generation scheduling problem of thermal units. Proceedings of 1998 International Conference on Power System Technology (POWERCON '98), the CSEE; Vol. 1. 18–21 Aug. 1998; p. 552–556.

14. Huang S-J. Enhancement of hydroelectric generation scheduling using ant colony system based optimization approaches. IEEE Trans Energy Conversion 2001; 16(3):296–301.

15. Shi L, Hao J, Zhou J, Xu G. Short-term generation scheduling with reliability constraint using ant colony optimization algorithm. Fifth World Congress on Intelligent Control and Automation, 2004 (WCICA 2004), the IEEE; Vol. 6. 15–19 June 2004; p. 5102–5106.

16. Sisworahardjo NS, El-Keib AA. Unit commitment using the ant colony search algorithm. Large Engineering Systems Conference on Power Engineering 2002 (LESCOPE '02), the IEEE; 26–28 June 2002; p. 2–6.

17. Sum-im T, Ongsakul W. Ant colony search algorithm for unit commitment. 2003 IEEE International Conference on Industrial Technology, the IEEE; Vol. 1. 10–12 Dec. 2003; p. 72–77.

18. Hou Y-H, Wu Y-W, Lu L-J, Xiong X-Y. Generalized ant colony optimization for economic dispatch of power systems. Proceedings of International Conference on Power System Technology, 2002 (PowerCon 2002), the CSEE; Vol. 1. 13–17 Oct. 2002; p. 225–229.

19. Abraham A, Ramos V. Web usage mining using artificial ant colony clustering and linear genetic programming. The 2003 Congress on Evolutionary Computation, 2003 (CEC '03), the IEEE; Vol. 2. 8–12 Dec 2003; p. 1384–1391.

20. Dorigo M, Bonabeau E, Theraulaz G. Ant algorithm and stigmergy. Future Generation Computer Systems 2000; 16(8):851–871.

21. Dorigo M. Optimization learning and natural algorithm. PhD dissertation; Politecnico di Milano, 1992.

22. Dorigo M, Maniezzo V, Colorni A. The ant system: Optimisation by a colony of co-operating agents. IEEE Trans Systems Man Cybernetics B 1996; 26(1):29–41.

23. Teng J-H, Liu Y-H. A novel ACS-based optimum switch relocation method. IEEE Trans Power Syst 2003; 18(1):113–120.

24. Jeon Y-J, Kim J-C, Yun S-Y, Lee KY. Application of ant colony algorithm for network reconfiguration in distribution systems. Proceedings, IFAC Symposium for Power Plants and Power Systems Control, Seoul, South Korea, The IFAC; June 9–12, 2003; p. 266–271.