IJCAT - International Journal of Computing and Technology, Volume 2, Issue 8, August 2015
ISSN : 2348 - 6090
www.IJCAT.org

Job Shop Scheduling Using ACO

Ms. K. Sathya Sundari

(Ph. D., Part time Category – B Research & Development Centre, Bharathiar University, Coimbatore)
Tamil Nadu, India

Abstract - Job shop scheduling using ACO(Ant Colony Optimization) approach. Different heuristic information is discussed and three different ant algorithms are presented. State transition rule and pheromone updating methods are given. The concept of the new strategy is highlighted and template for ACO approach is presented.

Keywords **– Job Shop Scheduling.**

1. Ant Colony Optimization

1.1 Behavior of Real Ants

Ants have a limited awareness of their surroundings. They are not able to see a food source that is available in a long way away and also are not having the knowledge to plan a correct route to reach there and come back. The ants work collectively to search the area around their nest. They need to communicate in some way so as to work collectively and this communication is carried out by marking the ground with pheromone. The nature of the ants varies considerably. Some types of ants are capable of finding the best route between a food source and the nest (Li and Gong 2003). A simplified theory of how ants find the best path to a food source is given in this thesis and key concepts relevant to ant algorithms are highlighted. All ants wander back and forth between the food source and the nest after finding the food source.

Initially, they take a route by a random wander between the two sites. They also deposit pheromone on the ground during the wandering. This pheromone will affect paths of future ants, because the ants have a tendency to follow paths marked with pheromone. If the pheromone level on a path is more, the more likely, an ant is to wander in that path. When ants first leave the nest, they choose paths randomly.

Ants that have returned from the nest using shorter routes will arrive more quickly to the nest and will deposit pheromone sooner on that path. That is, the shorter routes will be more strongly marked with pheromone and ants in future will be more likely to use those routes. After some time, the shorter routes become very strongly marked with pheromone and thus, all ants will be following the shorter routes.

Ants are capable of finding the shortest path from a food source to the nest (Dorigo and Gambardella 1997a, Colorni et al 1993) without using visual cues. Hence, the inspiring source of ACO is the “pheromone trail laying and following” behavior of real ants, which use pheromone (Holldobler and Wilson 1990, Beckers et al 1992) as a communication medium. The probability for choosing the next path by an ant will be directly proportional to the amount of pheromone on that path. Referring to Figure 4.1(a), three ants start from a point A, where nest is available, travel randomly to reach food at a point D and deposit some amount of pheromone during their traveling. First ant selects a path through point B. Second ant travels directly to D and third ant travels through C. In Figure 4.1(b), second ant reaches the food first, since it travels through the shortest path and other two ants are still traveling towards the food in their paths. At the same time, three more ants start to travel from the nest. Since the pheromone level for all paths traveled by ants are same, new ants follow the paths already traveled by the previous ants. Since the ants travel at constant speed, more number of ants travel through path A-D than number of ants through paths A-B-D and A-C-D as denoted in Figures 4.1(c) and 4.1(d). In turn, pheromone is deposited on the paths according to the number of ants traveling in the respective paths. Figure 4.1(e) shows thickness of the line indicating pheromone level of three paths after some time. Eventually, all ants choose the path A-D that is having the highest pheromone level.

(a) Ants Select Paths Randomly,

(b) Ant Selecting the Shortest Path Reaches

(c) and (d) Food First, Change of Pheromone Level

During Traveling,

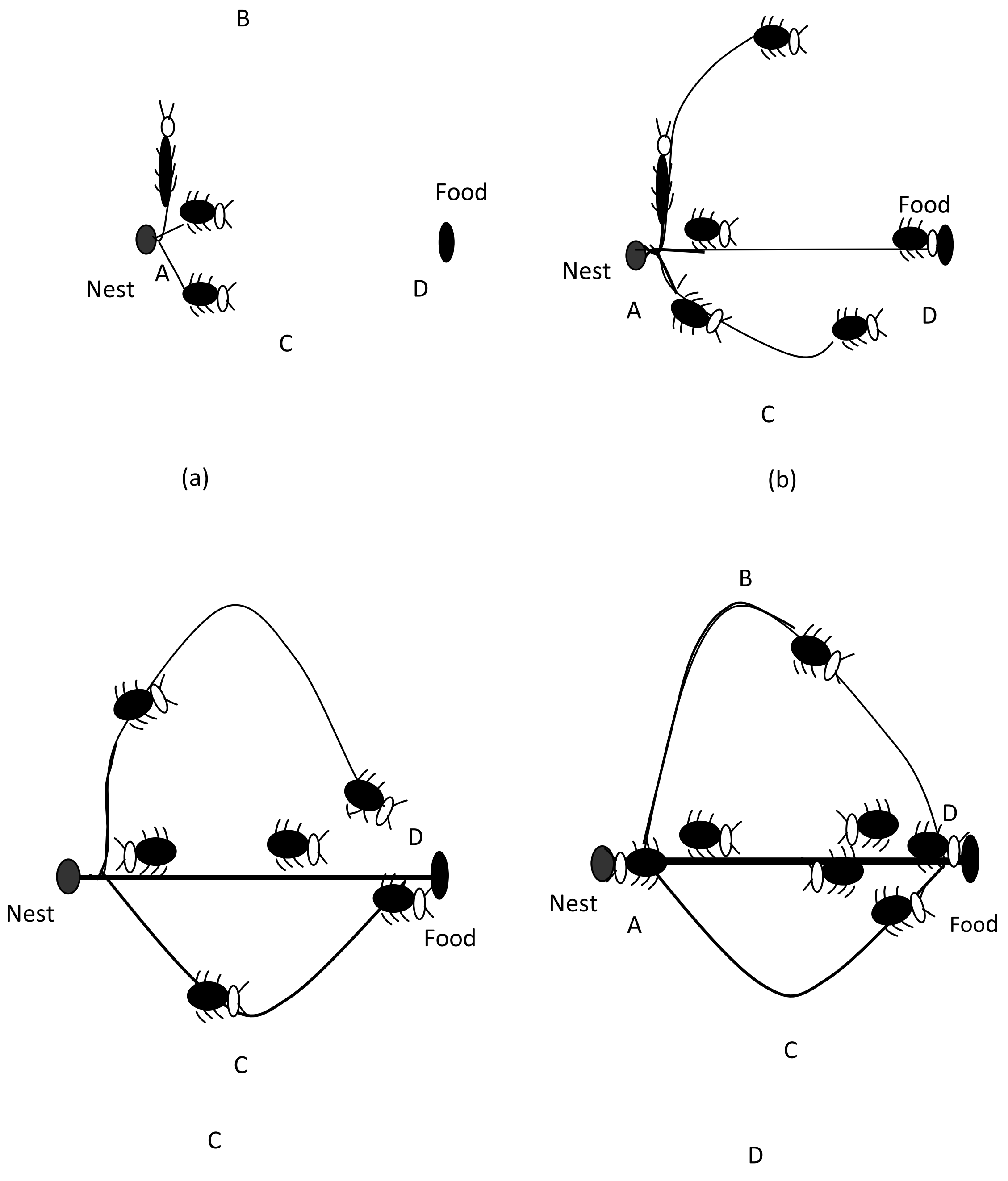
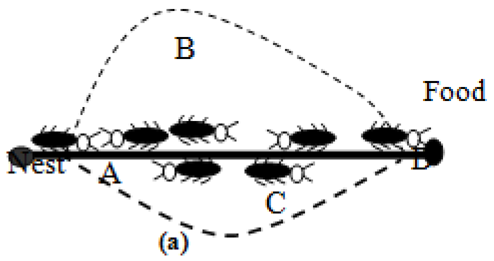


Figure 4.1 Pheromone Deposited in Search Space

Pheromone is deposited and evaporated in paths of ants during the traveling of ants. Since the paths A-B-D and A-C-D are not used by ants, the pheromone is evaporated on these paths as denoted in Figure 4.2(a).



(a) Pheromone Starting to Evaporate,

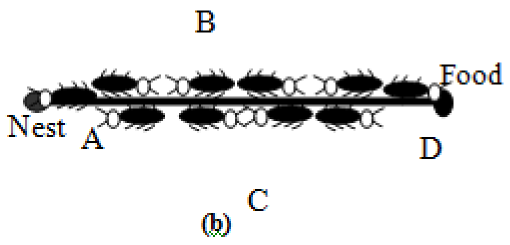


Figure 4.2 Pheromone Evaporation in Search Space

After some time the pheromone is completely evaporated in longer paths, which are not available long time and all ants as denoted in Figure 4.2(b) use only one path having the shortest distance.

Ants are also capable of adapting to changes in the environment. For example, the shortest path found so far is no longer feasible due to a new obstacle on the way. Figure 4.3 shows the behavior of real ants. Ants are moving on a straight line, which connects a food source to the nest (Figure 4.3(a)). As stated, each ant probabilistically prefers to follow a direction rich in pheromone rather than a poorer one. This elementary behavior of real ants can be used to explain how they can find the shortest path, which reconnects a broken line after the sudden appearance of an unexpected obstacle interrupting the initial path (Figure 4.3(b)). Once the obstacle has appeared, those ants, which are just in front of the obstacle, cannot continue to follow the pheromone trail and therefore they have to choose between turning right or left (Figure 4.3(c)). In this situation, half number of ants may be expected to turn right and the other half to turn left.

The same situation can be found on the other side of the obstacle. It is interesting to note that those ants, which choose, by chance, shorter path around the obstacle will more rapidly reconstitute the interrupted pheromone trail compared to those, which choose longer path. Hence, shorter path will receive a higher amount of pheromone in the time unit and this will in turn cause a higher number of ants to choose shorter path (Figure 4.3(d)). Due to this positive feedback, very soon all ants will choose the shorter path. The most interesting aspect of this process is that finding the shortest path around the obstacle seems to be an emergent property of interaction between the obstacle shape and distributed behavior of ants. Although all ants move at approximately the same speed and deposit a pheromone trail at approximately the same rate, it is a fact that it takes longer to contour obstacles on their longer side than on their shorter side, which makes the pheromone trail accumulate quicker on the shorter side. The ant's preference for higher pheromone trail levels makes this accumulation still quicker on the shorter path.

Generally, the real situation is much more complicated. The first ant to discover a food source may return to the nest along a wandering route. The current route will be improved over time towards some optimal route. Skaife (1961) describes a number of experiments on ant path-finding behavior. Some findings are relevant to this thesis. They are:

- If a new shorter route than the current route is found, ants will continue to use the older and longer route for some time, because ants tend to follow the heavily marked trail that they had previously been using.
- Ants have difficulty with paths having many sharp corners. The ants reach these corners and take some time finding the way to continue. They may follow the trail back again, instead of the trail onwards, since both are equally well marked.
- Objects marked with pheromone also affect the behavior of ants for some time after they are marked. However, the effect of trail diminishes, if the future ants will not deposit more pheromone.
- If the pheromone level between the food source and the nest changes, the ants will try to find a new path using any elements of the old trails, which are also relevant at the current time. This often greatly reduces the time taken to find a new path.

Key features need to be emphasized as follows:

- Ants co-operate by marking the search space they explore with pheromone.

- Ants wander at random, but are more likely to follow routes that have been marked with pheromone and the probability of following that route is greater, if the pheromone level is more.

2. Fundamentals of Ant Colony Optimization

ACO, is one of the meta-heuristic algorithms and first member of ACO is Ant System (AS), which was initially proposed by Dorigo et al (1991). AS uses a population of co-operating ants also known as agents (Colorni et al 1992). The cooperation phenomenon among the ants is called *foraging and recruiting* behavior (Dorigo et al 1996, Dorigo and Gambardella 1997b, Colorni et al 1991, Dorigo 1992). This describes how ants explore the world in search of food sources, then find their way back to the nest and indicate the food source to the other ants of the colony. The nature of ants, that collectively solve hard problems, gave rise to artificial ant algorithms. These algorithms were also proposed as a multi-agent approach in order to solve hard combinatorial optimization problems. ACO meta-heuristic introduces main features of artificial ants (Dorigo et al 1999) and these features have inspired different ant algorithms to solve hard optimization problems. ACO employs artificial ants in order to probabilistically construct a solution by iteratively adding solution components to partial solutions by using following information:

(i) Heuristic information on the problem instance being solved

(ii) Artificial pheromone trails, which change dynamically at run-time to reflect the search experience of ants.

A stochastic component allows ants to construct a variety of different solutions so as to explore much larger number of solutions, while the use of heuristic information, which is readily available for many problems, can direct the ants towards the most promising solutions. The solution constructions in future iterations of the algorithm are influenced by reminiscent of reinforcement learning (Sutton and Barto 1998). Additionally, the use of a colony of ants can give the algorithm-increased robustness and in many ACO applications, the collective interaction of a population of ants is needed to efficiently solve a problem and is able to increase the robustness of the algorithm. Application domain of ACO algorithms is vast. In principle, ACO can be applied to any discrete optimization problem, for which some solution construction mechanism can be conceived. These applications comprise two main application fields (Dorigo and Stutzle 2003).

- NP-hard problems where ACO algorithms are coupled with extra capabilities, such as heuristic information and local search methods.
- Dynamic optimization problems, in which some properties of the problem change over time concurrently with the optimization process that has to adapt to the problem's dynamics.

ACO algorithm is able to use the elements of previous solutions. It generates constructive low-level solution by randomizing the construction in a Monte Carlo way. Genetic algorithms (Holland 1975) suggest a Monte Carlo combination of different solution elements, whereas ACO defines probability distribution by previously obtained solution components. The way, in which the solution components and associated probabilities are defined, is depending upon the problem-specific. A set of computational concurrent and asynchronous agents (a colony of ants) move through states of the problem that correspond to partial solutions of the problem to be solved and incrementally construct a solution to the problem. The ant evaluates the solution and modifies the trail value on the components used in its solution during the construction phase. This trial information will direct the search of future ants. In addition, an ACO algorithm uses two more mechanisms:

(i) Trail evaporation

(ii) Daemon actions

An unlimited accumulation of trials over some component is avoided by trail evaporation mechanism, which decreases all trail values over time. Daemon actions implement centralized actions such as the invocation of a local optimization procedure or update of global information, which decides whether to bias the search process from a non-local perspective (Dorigo and Stutzle 2003). Partial solutions are seen as *states* in ACO, which iteratively constructs a solution. Each ant moves from one state to another corresponding to more complete partial solution. At each move, an ant computes a set of feasible states from its current state and moves to one of the feasible states by using Monte Carlo probability. The heuristic information and the trail level play major role in order to find the probability of moving from a current state to one of the feasible states. After constructing solutions by all ants, trails are updated by increasing or decreasing the level of trails corresponding to moves that were part of "good" or "bad" solutions, respectively.

3. Representation

A graph is employed to model a problem and the idea is to search the best path in a graph, through which artificial ants walk in order to find short paths. Therefore, a key point is to define the graph and a stochastic transition rule. Ants use the stochastic transition rule to choose their path with respect to constraints of the problem in order to ensure that ants only perform consistent paths (Solnon 2000). Ants follow a construction policy instead of arbitrary movement on the graph during the construction of a solution. Heuristic information may also use scheduling costs to make probabilistic decisions on how to move on the graph. More precisely, each ant of the colony has the following properties:

- It exploits the graph in order to search feasible schedules with minimum cost
- It uses a memory in order to store information about the path it followed so far.
- It can be assigned to start an operation and one or more termination conditions.
- It tries to move to any states in its feasible set of states during solution construction by applying a state transition rule, which is a function of locally available pheromone trails τ and heuristic values η, the ant's private memory storing its past history and the problem constraints.
- It stops the construction procedure, if at least one of the termination conditions is satisfied.
- It can update the pheromone trail, associated with a state (or with the corresponding connection), at the time of adding that state to current partial solution. This is called online step-by-step pheromone update or local pheromone update.
- It can retrace the same path backward and update the pheromone trails of the used components or connections after construction of a solution. This is called online delayed pheromone update or global pheromone update.

4. Heuristic Information

Heuristic information plays a major role to guide the ants during solution construction phase. It is also important, because it gives the possibility of exploiting problem specific knowledge. This information can be available a priori (static case) or at run-time (dynamic case). In static problems, the heuristic information is computed once at initialization time and it is considered same throughout the period of algorithm's run. On the other hand, in dynamic case, the heuristic information depends on the partial solution constructed so far and has to be computed at each step of an ant's movement. This causes a higher computational cost that may be compensated by higher accurateness of computed heuristic values. Some of the common heuristic information is given as follows:

- Earliest Due Date (EDD) – Sequence of due dates are used to rank unscheduled jobs. Highest rank value is given to jobs with earliest due dates. Problems without release and setup times prefer this type of heuristic information, which has the objective of minimizing completion time.
- Modified Due Date (MDD) – If due dates are past, sequence of due dates or completion time are used to rank unscheduled jobs. Jobs with the earliest due dates or completion time are given the highest rank. This type of heuristic information is used specifically for tardiness problems.
- Minimum Slack Time (MST) – The possible delay until the latest starting time is used to rank unscheduled jobs. Job with the smallest delay is given the highest rank.
- Earliest Completion Time (ECT) – Sequence of completion time is used to rank unscheduled jobs and the highest rank is given to a job with the earliest completion times. ECT is a dynamic heuristic, because it depends on the status of current schedule.
- Shortest Setup Time (SST) – Sequences of setup time are used to rank unscheduled jobs. Jobs with the shortest setup time are given the highest rank. This heuristic information is appropriate for problems without release times and has the objective of minimizing the completion time.
- Shortest Processing Time (SPT) – Sequences of processing time are used to rank unscheduled jobs and the highest rank is given to jobs with the shortest processing time. The processing time may be considered for an individual operation in a job or for a whole job.
- Job time/Operation time – Unscheduled jobs are ranked by the combination of total job time and an individual operation to be selected during the construction phase. An operation with the lowest processing time belonging to the highest job time is more desired.
- Job remaining time/Operation time – Here an operation with the lowest processing time belonging to the highest remaining job time is more desired.

There has been other ways of computing heuristic information (Dorigo and Stutzle 2003), viz. lower bound on solution cost of the completion of a partial schedule is used to compute the heuristic information. This method allows excluding certain choices, which lead to schedules that are worse than the best solution found so far. Combination of knowledge on the calculation of lower bound from mathematical programming with ACO paradigm is permitted in this case, but the computations of upper bounds are time consuming, because they have to be calculated in single step by each ant.

5. Ant System

Ant System (AS) is a first member of algorithms inspired by behavior of real ants and was proposed by Dorigo et al (1991). AS is being the prototype of a number of ant algorithms, which collectively implement ACO paradigm. As mentioned in section 4.1.3, a graph can be employed to represent the ant algorithm and in the graph, ants move along each branch from one node to another node and so construct paths representing solutions. Starting from an initial node, each ant chooses the next node in its path according to a state transition rule by using the probability of transition. Let S be a set of nodes at a decision point i. Transition probability for choosing the edge from i to node r by an ant at a time t is calculated as in Equation (4.1).

$$P(i,r)_t = \begin{cases} \dfrac{[\tau(i,r)]^{\alpha}.[\eta(r)]^{\beta}}{\sum_{j\in S}[\tau(i,j)]^{\alpha}.[\eta(j)]^{\beta}} & \text{if } r\in S \\ 0 & \text{otherwise} \end{cases} \tag{4.1}$$

$\tau(i,\ j)$ is the quantity of pheromone on the edge between node i and node j. $\eta(j)$ is heuristic information on the node j. α and β tune relative importance in probability of the amount of the pheromone and the heuristic information respectively. In case of JSSP, if the heuristic information refers to inverse of the operation time of the node j (i.e. $\eta(j) = 1/p_{ij}$), the artificial ants move to a node that has a higher amount of pheromone and lesser value of operation time and will have higher probability to be scheduled in the partial solution. After each iteration t of the algorithm, That is, when all ants have completed a solution, trails are updated by means of Equation (4.2).

$$\tau(i,\ j)_t = (1-\rho).\tau(i,\ j)_{t-1} + \Delta\tau(i,\ j) \tag{4.2}$$

where ρ $(0 < \rho < 1)$, is a user-defined parameter called *evaporation coefficient,* and $\Delta\tau(i, j)$ represents sum of the contributions of all ants that used move (i, j) to construct their solution and is calculated as given in

Equation (4.3). $$\Delta\tau(i, j) = \sum_{k=1}^{t} \Delta\tau(i, j)^k \tag{4.3}$$

where t is the total number of ants, $\Delta\tau(i,\ j)^k$ is the amount of trail laid on edge (i, j) by kth ant and is computed as in Equation (4.4),.

$$\Delta\tau(i,j)^k = \begin{cases} \dfrac{Q}{L_k} & \text{if ant } k \text{ uses arc } (i,j) \text{ in its tour} \\ 0 & \text{otherwise} \end{cases} \tag{4.4}$$

where Q is a constant parameter selected according to size of the problem and L_k is the length of the path traveled by kth ant. Hence, contributions of ants are proportional to the quality of solutions generated by corresponding ants and higher trail contributions are added to moves made by the ants giving better solutions. AS simply iterates a loop, where all ants construct in parallel their solutions, thereafter updating the trail levels. Performance of the algorithm depends on correct tuning of parameters like α, β, ρ, initial trail level τ_0, number of ants, and the constant Q used for generating high quality solutions with low cost.

6. Ant Colony System

AS was initially applied to solve traveling salesman problem. However, it was not able to compete against the state-of-the art algorithms in the field. Authors of AS, on the other hand, have the merit to introduce ACO algorithms and show the potentiality of using artificial pheromone and artificial ants to search of better solutions for complex optimization problems. Then research on AS was carried out in order to achieve the following goals:

- To improve the performance of the algorithm.
- To investigate and better explain the behavior of the algorithm.

Gambardella and Dorigo (1995) proposed an extension of AS called Ant-Q algorithm, which integrates several ideas from Q-learning (Watkins and Dayan 1992). They also proposed Ant Colony System (ACS)

(Gambardella and Dorigo 1996, Dorigo and Gambardella 1997b), which is a simplified version of Ant-Q. ACS maintains the same level of performance as Ant-Q algorithm in terms of the complexity and the computational results. The following three aspects make ACS to differ from AS: (i) State transition, (ii) Pheromone updating and (iii) Hybridization and performance improvement.

State Transition Rule

Ants use state transition rule to select the next state that is to be added to partial solution. ACS employs a transition rule called *pseudo-random-proportional*, which is a balance between *pseudo-random* state choice rule used in Q-learning (Watkins and Dayan 1992) and *random-proportional* action choice rule used in AS. In ACS, an ant selects a state using the biased random choice as in AS during some of the time, whereas the best state is selected during the rest of the time based on the heuristic information and the pheromone level. *Pseudo-random-proportional* rule selects the best state with a probability q_0 and selects a random state with a probability 1-q_0 where q_0 is a constant given as input ranging from 0 to 1. However, all the time, r*andom-proportional* rule used in AS selects the next state randomly with a probability distribution, which depends on the heuristic information and the pheromone level. *Pseudo-random-proportional* state transition rule in ACS provides a way to compromise between exploration of new states and exploitation of the heuristic information and the pheromone level. Hence, *pseudo-random-proportional* rule uses a state transition rule given in Equation (4.5).

$$s = \begin{cases} \underset{j \in S}{Max}\{[\tau(i,j)]^{\alpha}.[\eta(j)]^{\beta}\} & \text{if } q \leq q_0 \\ r & \text{otherwise} \end{cases} \quad (4.5)$$

where $q \in [0,1]$ is a uniform random number and r is a component, which is chosen randomly according to the probability distribution defined by Equation (4.1). The random number q is selected each time an ant moves from a state i to another state j. If the value of q is less than or equal to the value of q_0, the ant will select the best state. Otherwise, the ant will select a biased random state.

Pheromone updating

Once all ants have constructed solutions, AS updates the pheromone trail using all solutions generated by the colony of ants. An amount of pheromone on each edge belonging to one of the computed solutions is modified by an amount, which is proportional to its solution value. AS then evaporates the pheromone of the entire system after construction of solutions by all ants and the process of solution construction and pheromone update are iterated. But ACS updates pheromone value for the edges belonging to the best solution computed since the beginning of the computation and this technique is called global pheromone update. Global pheromone updating technique updates the amount of pheromone on edge (i, j) belonging to the shortest path at a time t by using the pheromone on that edge at the time t-1 as given in Equation (4.6).

$$\tau(i,j)_t = \begin{cases} (1-\rho).\tau(i,j)_{t-1} + Q/L_b & \text{if } (i,j) \in \text{Global best path} \\ \tau(i,j)_{t-1} & \text{otherwise} \end{cases} \quad (4.6)$$

where ρ is an evaporation co-efficient, Q is a constant whose value is chosen depending upon the problem size and L_b is length of the best path. Amount of pheromone deposited on each edge is inversely proportional to length of the path so as to enable shorter path to get higher amount of pheromone deposited on the edges. Global pheromone updating increases the attractiveness of promising solutions and tries to avoid long time of convergence by directly concentrating search of the best solution found up to the current iteration. ACS also employs a technique called local pheromone updating, which is intended to avoid a strong edge being chosen by all ants. While constructing its path, the local pheromone updating technique modifies amount of pheromone on the passed edge (i, j) at a time t as given in Equation (4.7).

$$\tau(i,j)_t = (1-\rho).\tau(i,j)_{t-1} + \rho.\tau_0 \quad (4.7)$$

where τ_0 is the initial amount of pheromone deposited on each edge and can be defined as $(n.L_{nn})^{-1}$ (Gambardella and Dorigo 1996), where L_{nn} is length of the path produced by SPT rule. Local pheromone updating modifies pheromone trail on the edges in each time the ant travels through these edges. Local pheromone updating also represents evaporation of the pheromone in natural ants and forgets previous good paths in favor of the new best path.

Structure of basic ant colony optimization algorithm

General structure of ACO algorithm is as follows.

```
Initialization
    Initialize parameters like $\alpha$, $\beta$, $\rho$, $q_0$, and $Q$.
    Store a maximum value to the solution.
    Calculate initial pheromone value
Construction and Improvement
    While termination condition not satisfied do
Construction
    For each ant $k$ do
    Choose a state $i$ with a probability and add $i$ to partial solution
     Update pheromone trial for the current move
      End For
    Update pheromone trail for the best ant's path
    Improve the solution
    End While
Output
     Print the best solution found.
```

he algorithm initializes various parameters and assigns a maximum value for the current solution. It also calculates initial pheromone value during the initialization phase. In the construction phase, the algorithm finds a solution and updates the pheromone trial, which is used to improve the solution. The algorithm repeats the construction phase until termination condition is met and finally prints the best solution found.

Performance improvement by hybridization

ACS incorporates an advanced data structure known as *candidate list* (Reinelt 1994) in order to solve big symmetric and asymmetric traveling salesman problems (TSP/ATSP) (Gambardella and Dorigo 1996, Dorigo and Gambardella 1997b). A static data structure is used to implement the candidate list, which has length l and contains l preferred cities to be visited from a given city i. In ACS, the state transition rule is used to select a city in the candidate list. If the candidate list is empty (i.e. none of the cities are available in the candidate list), the ant chooses the nearest available city only using the heuristic value η_{ij}. Performance of ACS for TSP/ATSP has been improved by incorporating local optimization heuristic and this technique is known as *hybridization*, in which a solution generated by the ant is taken to its local minimum by the application of a local optimization heuristic. ACS considers new optimized solutions as final solutions produced in the current iteration by the ants and uses these optimized solutions to globally update the pheromone trails. This ACS implementation, which has the combination of a new pheromone management policy, a new state transition strategy and local search procedures, was finally competitive with state-of-the-art algorithm for solving TSP/ATSP problems. This kind of implementation opened a new frontier for ACO based algorithm. ACO algorithms were able to break several optimization records, including those for job shop scheduling and routing problems by employing the approach that combines a constructive phase driven by the pheromone and local search phase, which optimizes the constructed solution.

Max-Min Ant System

Relatively high pheromone levels on certain solution components may quickly lead to those components being used and causes the exclusion of all other solution components. It results in premature convergence to a local optimal solution. Due to pheromone saturation of existing longer routes, a colony is unable to exploit new and efficient solution components. This phenomenon was especially noticeable on large problem instances and motivates the development of Max-Min Ant System (MMAS) (Stutzle and Hoos 1996, Stutzle and Hoos 1998, Stutzle and Hoos 2000).

Distinguishing characteristic of MMAS is that it maintains pheromone values within the range [τ_{min}, τ_{max}] at all time (Blum and Sampels 2004). Pheromone levels are initially set to τ_{max} to encourage exploration early in the search process (Stutzle and Hoos 2000). It was found that the best solutions were found at the time of stagnation and these solutions are used to guide the selection of values for τ_{min} and τ_{max}. This guidance sets the appropriate bounds, which are independent of the problem specific. MMAS was improved by using a static candidate sets and the addition of a local search heuristics. Diversification mechanism is also used in MMAS to force the discovery of new solutions that are far from the global-best solution. The diversification mechanism reinitializes all pheromone values to τ_{max}, if small change is detected on solutions generated over time. MMAS has made a significant success across a range of problems.

7. Conclusion

This chapter has provided meta-heuristic approach called ant colony optimization. Behavior of ants to find shortest path has been given. Different ant algorithms have been discussed together with local and global pheromone updating. The key to the application of ACO to a new problem is to identify an appropriate representation for the problem (to be represented as a graph searched by many artificial ants), and an appropriate heuristic that defines the distance between any two nodes of the graph. Then the probabilistic interaction among the artificial ants mediated by the pheromone trail deposited on the graph edges will generate good, and often optimal, problem solutions. Other problems solved by ACO algorithms include: graph partitioning; subset problems including knapsack problems; Quadratic assignment; graph colouring; vehicle routing; networking routing and many more.

References

[1] M. Dorigo & L. M. Gambardella, 1997. "Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem". IEEE Transactions on Evolutionary Computation, 1 (1): 53–66.

[2] M. DorigoT. Stützle,“The Ant Colony optima-zation Metaheuristic: Algorithms, Applications, & Advances”, Handbook of Metaheuristics, 2002

[3] M. Dorigo et L.M. Gambardella, *Ant Colony System : A Cooperative Learning Approach to the Traveling Salesman Problem*, IEEE Transactions on Evolutionary Computation, volume 1, numéro 1, pages 53-66, 1997.

[4] A. Colorni, M. Dorigo et V. Maniezzo, *Distributed Optimization by Ant Colonies*, actes de la première conférence européenne sur la vie artificielle, Paris, France, Elsevier Publishing, 134-142, 1991.

[5] M. Dorigo and T. Stützle. *Ant Colony Optimization*. MIT Press, Cambridge, MA, 2004.