Ant Colony Optimisation (ACO)

Timo Petmanson
University of Tartu
11 May 2011

Abstract

Ant colony optimisation (ACO) among with swarm intelligence methods is a new and promising area of research in artificial intelligence. In this work, we explore the origins of ACO, some most effective algorithms utilising it and shed some light on the direction research is going in the area.

1 Introduction

The ant colony optimization (ACO) problem solving methods are directly inspired by the foraging behavior of some species of ants. In an experiment carried out by Deneubourg [DAGP90], the colonies self-organized themselves effectively to accommodate to the specifics of the environment, making the access to the food source optimal. The general idea of ACO is that each ant leaves a trace of pheromone along the path it moves. The chemical is recognized by other ants and paths having the highest concentration of pheromone are more likely to be followed [DBS06]. This has lead to a mathematical formalization of ACO and implementations to many common NP-complete problems such as the traveling salesman problem, graph coloring etc.

2 Optimization technique

To formalize the ACO, we need the general combinatorial optimization problem model $P$, which is a tuple $(S, \Omega, f)$, where $S$ is the search space defined over finite set of discrete decision variables $X_i$, $i = 1 \ldots n$. Each of these variables can be assigned a value from set $D_i = \{v_i^1, \ldots, v_i^{|D_i|}\}$, forming a solution component $c_{ij} \in \cup D_i = C$. The complete enumeration of these assignments over variable form the total search space. An objective function $f : S \rightarrow \mathbb{R}^+$ is the one that has to be minimized, defining the fitness of a particular solution $X_1X_2\ldots X_n$, where $X_i = c_{ij} \in C$, $i = 1 \ldots n$ [DC99].

One single feasible solution is constructed by a single artificial ant. The ant is given a construction graph, which is fully connected and each vertex or edge represent a partial solution $c_{ij} \in C$, thus traversing the graph either by vertices or edges leads to a full solution. A pheromone value $\tau_{ij}$ is associated with each solution component $c_{ij}$ and the ant can additionally increase the pheromone by some amount on the components. That may also depend on the quality of the solution.

The overall process is described by Algorithm 1 [DCG99]. After initialization, we iterate as long as we get a good-enough solution. Step ConstructAntSolutions takes $m$ artificial ants and lets them to build $m$ solutions. Each solution component is guided by a stochastic mechanism, which takes into account the amount of pheromone on the other components of the partial solution. Next step, ApplyLocalSearch is meant to improve the results returned by ants. Although this step is optional, it is mostly used in advanced ACO algorithms. Step UpdatePheromones usually increases the levels of components belonging to good solutions and decreases levels for bad solutions to make ants more likely to avoid them.

3 Main ACO Algorithms

3.1 Ant System (AS)

Ant System was the first algorithm proposed in the area. Its main property is that the pheromone values are updated by all ants that have built a solution in a single iteration. Basically after each iteration, we update pheromone like this:

$$\tau_{ij} = (1 - p)\tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^k$$

where $m$ is the number of ants and $p$ is the evaporation rate. The $\Delta\tau_{ij}^k$ is some constant value divided by the
length of the tour constructed by an ant $k$. The ant chooses next solution component by evaluating $\tau_{ij}^a \eta_{ij}^b$, where $\eta_{ij}$ is the heuristic information. Constants $\alpha$ and $\beta$ control the importance of pheromone vs the heuristic.

The heuristic is something very problem specific. In case of traveling salesman problem, it could be $1/d$, where $d$ is distance between the possible next town to be visited [DBS06]. For example, the probability of going to city $j$ from city $i$ is

$$p_{ij}^k = \begin{cases} \tau_{ij}^a \eta_{ij}^b / \sum_{c_{ij} \in N} \tau_{il}^a \eta_{il}^b & c_{ij} \in N \\ 0 & \text{otherwise} \end{cases}$$

(2)

where $N$ is a set of feasible components when ant $k$ is in city $i$.

### 3.2 Max-Min Ant System (MMAS)

The MMAS takes into account only the best solution produced by some ant. The pheromone update is similar to Equation (1), but only using the amount of best ant instead of all of them. The new pheromone values only exist on edges of the tour of the best ant. Other edges are given pheromone amount of zero. Additionally, the outcome is bounded, thus the pheromone amount cannot get lower and neither higher than some predefined bounds $\tau_{max}$ and $\tau_{low}$ [DBS06]. Thus

$$\tau_{ij} = \min(\tau_{max}, \max(\tau_{low}, (1 - p)\tau_{ij} + \Delta\tau_{ij}^{best}))$$

(3)

where

$$\Delta\tau_{ij}^{best} = \begin{cases} 1/L^{best} & \text{if } i,j \text{ belongs to the best tour} \\ 0 & \text{otherwise} \end{cases}$$

$L^{best}$ is the length of the tour of the best ant.

### 3.3 Ant Colony System (ACS)

The main difference with ACS and AS, is that former has a local pheromone update step. That is, after each construction step, the ant decreases the pheromone value of last traversed edge (or vertex). This encourages following ants to choose different edges and produce more different solutions. The offline update step is similar to MMAS, where the values of the best ants are used, but with difference that the other edges are not assigned zero values, but are left as they were.

Another difference between AS is the decision rule. At first, by some probability, the ant chooses the way mainly by the amount of pheromone, otherwise it works as AS given with Equation (2).

### 4 Future of ACO

The current emphasis of the research is mainly still on the applications. Namely, how to apply ACO to more and more academic, but even more on problems arising in industry. For example, how to make ACO work in a dynamic environment, which changes in time. One example are telecommunication networks, where different nodes may be online at different times or where failures may occur. Or how to solve stochastic traveling salesman problem. Given a probability that a city needs to be visited, how to cope with that. Also, research is carried out on how to parallelize the ACO methods effectively and how to make the methods work with multiple objectives [DBS06].

### Conclusions

As discussed in this paper, ant colony optimization methods are relatively new in the field of artificial intelligence and computer science in general. Still, they have drawn quite some attention as current implementations of some common problems are able to provide results as good as or better than alternatives, but with less computing power. The future research is much focused on making more industry-level problems approachable by ACO and making them better scalable to multiprocessing.

### References


