**Ant Colony Based Search**

The **ant colony optimization** [algorithm](http://en.wikipedia.org/wiki/Algorithm) (ACO) is a [probabilistic](http://en.wikipedia.org/wiki/Probability) technique for solving computational problems which can be reduced to finding good paths through [graphs](http://en.wikipedia.org/wiki/Graph_(mathematics)).

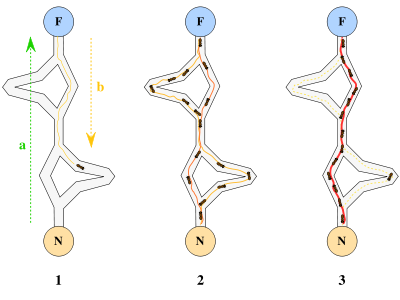
This algorithm is a member of **ant colony algorithms** family, in [swarm intelligence](http://en.wikipedia.org/wiki/Swarm_intelligence) methods, and it constitutes some [metaheuristic](http://en.wikipedia.org/wiki/Metaheuristic) optimizations. Initially proposed by [Marco Dorigo](http://en.wikipedia.org/wiki/Marco_Dorigo) in 1992 in his PhD thesis , the first algorithm was aiming to search for an optimal path in a graph; based on the behavior of [ants](http://en.wikipedia.org/wiki/Ants) seeking a path between their [colony](http://en.wikipedia.org/wiki/Ant_colony) and a source of food. The original idea has since diversified to solve a wider class of Numerical problems, and as a result, several problems have emerged, drawing on various aspects of the behavior of ants.

In the real world, ants (initially) wander [randomly](http://en.wikipedia.org/wiki/Random), and upon finding food return to their colony while laying down [pheromone](http://en.wikipedia.org/wiki/Pheromone) trails. If other ants find such a path, they are likely not to keep travelling at random, but to instead follow the trail, returning and reinforcing it if they eventually find food.

Over time, however, the pheromone trail starts to evaporate, thus reducing its attractive strength. The more time it takes for an ant to travel down the path and back again, the more time the pheromones have to evaporate. A short path, by comparison, gets marched over faster, and thus the pheromone density remains high as it is laid on the path as fast as it can evaporate. Pheromone evaporation has also the advantage of avoiding the convergence to a locally optimal solution. If there were no evaporation at all, the paths chosen by the first ants would tend to be excessively attractive to the following ones. In that case, the exploration of the solution space would be constrained.

Thus, when one ant finds a good (i.e., short) path from the colony to a food source, other ants are more likely to follow that path, and [positive feedback](http://en.wikipedia.org/wiki/Positive_feedback) eventually leads all the ants following a single path. The idea of the ant colony algorithm is to mimic this behavior with "simulated ants" walking around the graph representing the problem to solve.

***Detailed***

[](http://en.wikipedia.org/wiki/File:Aco_branches.svg)

The original idea comes from observing the exploitation of food resources among ants, in which ants’ individually limited cognitive abilities have collectively been able to find the shortest path between a food source and the nest.

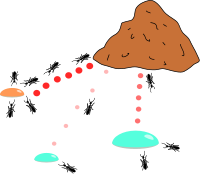
1. The first ant finds the food source (F), via any way (a), then returns to the nest (N), leaving behind a trail pheromone (b)
2. Ants indiscriminately follow four possible ways, but the strengthening of the runway makes it more attractive as the shortest route.
3. Ants take the shortest route, long portions of other ways lose their trail pheromones.

In a series of experiments on a colony of ants with a choice between two unequal length paths leading to a source of food, biologists have observed that ants tended to use the shortest route. A model explaining this behavior is as follows:

1. An ant (called "blitz") runs more or less at random around the colony;
2. If it discovers a food source, it returns more or less directly to the nest, leaving in its path a trail of pheromone;
3. These pheromones are attractive, nearby ants will be inclined to follow, more or less directly, the track;
4. Returning to the colony, these ants will strengthen the route;
5. If two routes are possible to reach the same food source, the shorter one will be, in the same time, traveled by more ants than the long route will
6. The short route will be increasingly enhanced, and therefore become more attractive;
7. The long route will eventually disappear, pheromones are volatile;
8. Eventually, all the ants have determined and therefore "chosen" the shortest route.

Ants use the environment as a medium of communication. They exchange information indirectly by depositing pheromones, all detailing the status of their "work". The information exchanged has a local scope, only an ant located where the pheromones were left has a notion of them. This system is called "[Stigmergy](http://en.wikipedia.org/wiki/Stigmergy)" and occurs in many social animal societies (it has been studied in the case of the construction of pillars in the nests of termites). The mechanism to solve a problem too complex to be addressed by single ants is a good example of a self-organized system. This system is based on positive feedback (the deposit of pheromone attracts other ants that will strengthen it themselves) and negative (dissipation of the route by evaporation prevents the system from thrashing). Theoretically, if the quantity of pheromone remained the same over time on all edges, no route would be chosen. However, because of feedback, a slight variation on an edge will be amplified and thus allow the choice of an edge. The algorithm will move from an unstable state in which no edge is stronger than another, to a stable state where the route is composed of the strongest edges.

Application

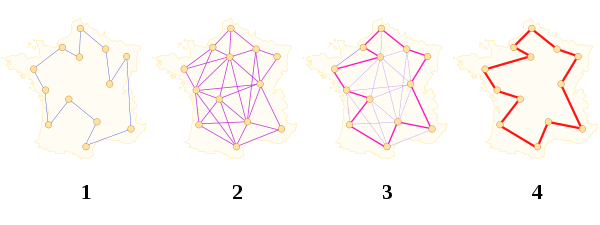
[](http://en.wikipedia.org/wiki/File:Knapsack_ants.svg)

[Knapsack problem](http://en.wikipedia.org/wiki/Knapsack_problem). The ants prefer the smaller drop of honey over the more abundant, but less nutritious, sugar.

Ant colony optimization algorithms have been applied to many combinatorial optimization problems, ranging from quadratic assignment to fold [protein](http://en.wikipedia.org/wiki/Protein) or [routing vehicles](http://en.wikipedia.org/w/index.php?title=Routing_vehicles&action=edit&redlink=1) and a lot of derived methods have been adapted to dynamic problems in real variables, stochastic problems, multi-targets and parallel implementations. It has also been used to produce near-optimal solutions to the [travelling salesman problem](http://en.wikipedia.org/wiki/Travelling_salesman_problem). They have an advantage over [simulated annealing](http://en.wikipedia.org/wiki/Simulated_annealing) and [genetic algorithm](http://en.wikipedia.org/wiki/Genetic_algorithm) approaches of similar problems when the graph may change dynamically; the ant colony algorithm can be run continuously and adapt to changes in real time. This is of interest in [network routing](http://en.wikipedia.org/wiki/Network_routing) and urban transportation systems.

As a very good example, ant colony optimization algorithms have been used to produce near-optimal solutions to the travelling salesman problem. The first ACO algorithm was called the Ant system [[5]](http://en.wikipedia.org/wiki/Ant_colony_optimization#cite_note-Ant_system-4) and it was aimed to solve the travelling salesman problem, in which the goal is to find the shortest round-trip to link a series of cities. The general algorithm is relatively simple and based on a set of ants, each making one of the possible round-trips along the cities. At each stage, the ant chooses to move from one city to another according to some rules:

1. It must visit each city exactly once;
2. A distant city has less chance of being chosen (the visibility);
3. The more intense the pheromone trail laid out on an edge between two cities, the greater the probability that that edge will be chosen;
4. Having completed its journey, the ant deposits more pheromones on all edges it traversed, if the journey is short;
5. After each iteration, trails of pheromones evaporate.

[](http://en.wikipedia.org/wiki/File:Aco_TSP.svg)

**"An example's Pseudo-code and formulas"**

procedure ACO\_MetaHeuristic

while(not\_termination)

generateSolutions()

pheromoneUpdate()

daemonActions()

end while

end procedure

**Edge Selection:**

An ant will move from node *i* to node *j* with probability


p_{i,j} = 
\frac
{ (\tau_{i,j}^{\alpha}) (\eta_{i,j}^{\beta}) }
{ \sum (\tau_{i,j}^{\alpha}) (\eta_{i,j}^{\beta}) }

where

τ*i*,*j* is the amount of pheromone on edge *i*,*j*

α is a parameter to control the influence of τ*i*,*j*

η*i*,*j* is the desirability of edge *i*,*j* (a priori knowledge, typically 1 / *di*,*j*)

β is a parameter to control the influence of η*i*,*j*

**Pheromone Update**

τ*i*,*j* = (1 − ρ)τ*i*,*j* + Δτ*i*,*j*

where

τ*i*,*j* is the amount of pheromone on a given edge *i*,*j*

ρ is the rate of pheromone evaporation

and Δτ*i*,*j* is the amount of pheromone deposited, typically given by


\Delta \tau^{k}_{i,j} = 
\begin{cases}
1/L_k & \mbox{if ant }k\mbox{ travels on edge }i,j \\
0 & \mbox{otherwise}
\end{cases}

where *Lk* is the cost of the *k*th ant's tour (typically length).

**Common extensions**

Here is some of most popular variations of ACO Algorithms

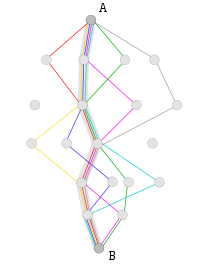
1. Elitist Ant System
   * The global best solution deposits pheromone on every iteration along with all the other ants
2. Max-Min Ant System (MMAS)
   * Added Maximum and Minimum pheromone amounts [τmax,τmin]
   * Only global best or iteration best tour deposited pheromone
   * All edges are initialized to τmax and reinitialized to τmax when nearing stagnation.
3. proportional pseudo-random rule. it has presented above
4. Rank-Based Ant System (ASrank)
   * All solutions are ranked according to their fitness. The amount of pheromone deposited is then weighted for each solution, such that the more optimal solutions deposit more pheromone than the less optimal solutions

For some versions of the algorithm, it is possible to prove that it is convergent (ie. it is able to find the global optimum in a finite time). The first evidence of a convergence ant colony algorithm was made in 2000, the graph-based ant system algorithm, and then algorithms for ACS and MMAS. Like most [metaheuristics](http://en.wikipedia.org/wiki/Metaheuristic), it is very difficult to estimate the theoretical speed of convergence. In 2004, Zlochin and his colleagues[[8]](http://en.wikipedia.org/wiki/Ant_colony_optimization#cite_note-Zlochin_model-based_search-7) have shown COA type algorithms could be assimilated methods of [stochastic gradient descent](http://en.wikipedia.org/wiki/Stochastic_gradient_descent), on the [cross-entropy](http://en.wikipedia.org/wiki/Cross-entropy) and [Estimation of distribution algorithm](http://en.wikipedia.org/wiki/Estimation_of_distribution_algorithm). They proposed that these [metaheuristics](http://en.wikipedia.org/wiki/Metaheuristic) as a "[research-based model](http://en.wikipedia.org/w/index.php?title=Research-based_model&action=edit&redlink=1)".

**Other examples**

The ant colony algorithm was originally used mainly to produce near-optimal solutions to the travelling salesman problem and, more generally, the problems of [combinatorial optimization](http://en.wikipedia.org/wiki/Combinatorial_optimization). It is observed that since it began its use has spread to the areas of [classification](http://en.wikipedia.org/wiki/Classification) and [image processing](http://en.wikipedia.org/wiki/Image_processing).

A difficulty in definition

[](http://en.wikipedia.org/wiki/File:Aco_shortpath.svg)

With an ACO algorithm, the shortest path in a graph, between two points A and B, is built from a combination of several paths. It is not easy to give a precise definition of what algorithm is or is not an ant colony, because the definition may vary according to the authors and uses. Broadly speaking, ant colony algorithms are regarded as [populated](http://en.wikipedia.org/wiki/People) [metaheuristics](http://en.wikipedia.org/wiki/Metaheuristics) with each solution represented by an ant moving in the search space. Ants mark the best solutions and take account of previous markings to optimize their search. They can be seen as [probabilistic](http://en.wikipedia.org/wiki/Probabilistic) [multi-agent](http://en.wikipedia.org/wiki/Multi-agent) algorithms using a [probability distribution](http://en.wikipedia.org/wiki/Probability_distribution) to make the transition between each [iteration](http://en.wikipedia.org/wiki/Iteration). In their versions for combinatorial problems, they use an iterative construction of solutions. According to some authors, the thing which distinguishes ACO algorithms from other relatives (such as algorithms to estimate the distribution or particle swarm optimization) is precisely their constructive aspect. In combinatorial problems, it is possible that the best solution eventually be found, even though no ant would prove effective. Thus, in the example of the Travelling salesman problem, it is not necessary that an ant actually travels the shortest route: the shortest route can be built from the strongest segments of the best solutions. However, this definition can be problematic in the case of problems in real variables, where no structure of 'neighbours' exists. The collective behaviour of [social insects](http://en.wikipedia.org/wiki/Social_insects) remains a source of inspiration for researchers. The wide variety of algorithms (for optimization or not) seeking self-organization in biological systems has led to the concept of "[swarm intelligence](http://en.wikipedia.org/wiki/Swarm_intelligence)", which is a very general framework in which ant colony algorithms fit.

Stigmergy algorithms

There is in practice a large number of algorithms claiming to be "ant colonies", without always sharing the general framework of optimization by canonical ant colonies (COA). In practice, the use of an exchange of information between ants via the environment (a principle called "[Stigmergy](http://en.wikipedia.org/wiki/Stigmergy)") is deemed enough for an algorithm to belong to the class of ant colony algorithms. This principle has led some authors to create the term "value" to organize methods and behavior based on search of food, sorting larvae, division of labour and cooperative transportation.