Ant Colony Optimization

Jeevesh Juneja

November 30, 2020

1 The Problem Domain

The ant colony optimization techniques are designed for computational problems that can be reduced to finding shortest paths in weighted graphs. Note here, that these paths can be constrained in various ways, like they must lead to a particular destination, or they may be constrained to be Hamiltonian paths or a Hamiltonian Cycle, or in any general way.

2 Principle

Now that the problem is set, we discuss the basic principle of ant colony optimization approaches. These approaches associate with each edge of the graph, a **Pheromone Level**, which are just simple numbers that the algorithm can use. These optimization techniques are based on the fact that if we have many small agents, each of which:

- Tries to find the solution to the problem by traversing the graph according to some probabilistic rule.
- Once **all** the agents have found their solutions, they all update the Pheromone Level of all edges they encountered on their path.

where,

- The Pheromone Levels of all edges are updated in accordance to the difficulty[1] of traversing that edge.
- The rule used to traverse must try to **locally** minimize the difficulty of finding the entire solution by observing Pheromone Levels[2] of outgoing edges.

then, we are guaranteed that all the small agents would end up following the optimal path through the graph, eventually.

3 Some Improvements

Now we will develop two innovative ideas to make our training better. Suppose, we initialise all pheromone levels, τ, to zero in the beginning(indicating all edges look equally difficult, initially), then after the all the small agents have found some solution, the τ of each **traversed** edge is updated, let's say by adding some little factor to each edge's previous pheromone level.

Then, although these traversed edges, may not be optimal, they still have higher pheromone levels than the edges that were not visited(which may be optimal). So, although with enough iterations, the optimum solution will still be found[3]; our optimization procedure has been slowed. To resolve this, we use an **evaporation factor,** ρ, a factor by which the pheromone level of an edge will decay after each iteration.

[1] for example, the length of the edge can be used as a measure of difficulty of traversing it, the longer the edge, the lesser its pheromone level will be increased

[2] which, now, indicate difficulty of locally seen edges, based on previous travels through the graph

[3] as we use a probabilistic rule for traversing the graph

This will cause the pheromones on the edges to evaporate with iterations. So, if at each iteration enough small agents don't choose that edge and deposit pheromones on it, the edge will go back to zero pheromone level, eventually.

The second idea, is to include a factor, η_{xy} that can capture our initial suppositions(if we have any) regarding which edge (x, y) is to be preferred. This information can help guide the optimization, and make it faster. Note here, that we can't include this information in the initial values of the pheromone levels, as then this information will be lost as pheromone levels decay.

Now, the things we have to decide are the update rules for Pheromone Levels and the local strategy for finding the least difficult outgoing edge.

4 Mathematical Formulation

Following presents the algorithm for the basic Ant Colony optimization. Note here how the change in pheromone level is inversely proportional to the length of the path. The parameter Q can be varied to control the strength of pheromones. The cost of the agent's solution in step (7) of Algorithm 1, is usually set as the sum of weights of all edges in the solution.

Algorithm 1: General Ant Colony Optimization

1: Initialize the graph $G = (V, E)$
2: Initialize the pheromone levels $\tau_{xy}^{(0)} = 1, \forall (x, y) \in E$
3: Initialize the evaporation rate $\rho = 0.1$, a constant $Q = 5$
4: Initialize the attractiveness for each edge η_{xy} according to problem.
5: **while** the small agents find different solutions **do**
6: Let all small agents find solution paths, following small agent strategy.
7: $L_k \leftarrow$ the cost of k-th small agent's solution
8: Set the change in pheromone level, $\Delta\tau_{xy}^k = \begin{cases} Q/L_k & \text{if small agent } k \text{ uses edge } (x, y) \text{ in its solution} \\ 0 & \text{otherwise} \end{cases}$
9: Update the pheromone levels $\tau_{xy} \leftarrow (1 - \rho)\tau_{xy} + \sum_k \Delta\tau_{xy}^k$ [4]
10: **end while**

Now we present a probabilistic small agent strategy, that is used for traversing the graph and finding solutions. The transitions are chosen with probabilities proportional to the product of their pheromone levels and η_{xy} .

Algorithm 2: Small Agent Strategy

1: **while** solution not found **do**
2: Find the set of allowed transitions from current node.
3: Calculate probability of each transition as $p_{xy} = \frac{(\tau_{xy})(\eta_{xy})}{\sum_{z \in \text{allowed}_x} (\tau_{xz})(\eta_{xz})}$
4: Make a transition according to these probabilities
5: **end while**

In the above algorithm, sometimes two extra hyper-parameters α and β are used as powers of all τ's and η's respectively. These hyper-parameters allow us to modulate the relative importance of the two factors(τ, η) that influence the probability.

5 Applying ACO to TSP

Suppose we want to solve the Travelling Salesman Problem for the graph in Figure 1. The first question we want to answer is, do we have any prior knowledge about our problem? How should we include it in η?

[4] where the summation is over all solutions found by small agents in the current iteration.

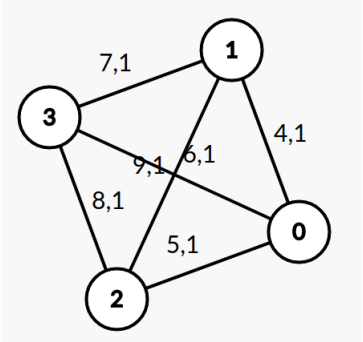

Figure 1: The first number indicates the length of the edge and the second number indicates the pheromone level. The edge between 1 and 2 has length 9.

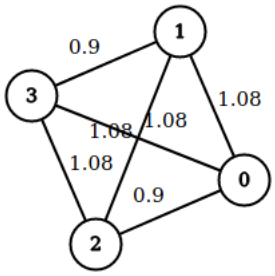

Figure 2: The pheromone levels after one iteration

Surely, we know here, that we should be preferring shorter paths! So.. let's set $\eta_{xy} = \frac{1}{d_{xy}}$ where d_{xy} is the length of edge between x and y. Let's go through a few iterations of the above algorithms with this setting of η.

Let's say we only have one small agent, which will go looking for a Hamiltonian cycle, and finds $(0 \to 1 \to 2 \to 3 \to 0)$ as the solution. Note here the allowed transitions at each step were only to those nodes, which weren't previously visited. The cost of the path, L_k would be $4+6+8+9 = 27$. So applying updates as in step (9) of the algorithm 1, the Figure 2 shows the updated pheromone levels. Note how even the pheromone levels of edges that weren't visited decayed to 0.9 from 1. Also, observe that irrespective of the length of the edge length, the pheromone levels of all traversed edges was raised by same amount. The length information was already taken care of in η.

The above iteration was done with $Q = 5$. If you now calculate the probability of transitions after reaching 1 from the path $0 \to 1$, you'll see that the transition $1 \to 2$ has higher probability than $1 \to 3$, even though $1 \to 2$ is longer. Had Q been smaller, this wouldn't have been the case. Also, had we used more than 1 small agents, possibly this wouldn't have been case just after first iteration. So, the parameter Q must be chosen according to the number of small agents we run parallel-ly, in one iteration.

Similarly, you can carry out further iterations. And finally reach the optimal solution of the TSP problem.