CS325 Final Project Report

Nathan, Newton, Ziwei

June 6, 2018

Research Summary

Ants Colony Algorithm is an interesting algorithm that takes inspiration from nature. When there is a food source available for ants to forage. The ants are able to find the shortest path to the source over time. The mechanism is as the follows: at the beginning, each path leads to the food source has an equal probability of being traversed by the ants. As ants traverse the path, they leave trails of pheromone. As ants find their way back home, the shortest path would allow the ants to return quicker. The new patch of ants leaving from the starting point would have a higher probability to traverse the path with the higher density of pheromone. Over time, the shortest path would have the highest density of pheromone, and the less often traveled path would have pheromone slowly decayed. Eventually, all ants would only traverse via the shortest path to the food source. This process can be considered as a positive feedback loop. The shortest path is reinforced over time.
Now let's apply this idea to TSP. Given a TSP with n cities and with distance d_{ij}. We distribute the ants randomly to each city. Let's assume that ants have memory and can remember what cities they have visited and would not visit them again. The ants would prefer to travel to the closer cities. The probability that a city j selected by ant k after city i is,

$$p_{ij}^k = \begin{cases} \frac{[\tau_{ij}]^\alpha \cdot [\eta_{ij}]^\beta}{\sum_{S \in U\ k} [\tau_{iS}]^\alpha \cdot [\eta_{is}]^\beta} & j \in U_k \\ 0 & otherwise \end{cases} \tag{1}$$

τ_{ij} is the intensity of pheromone between city i and city j. α is the parameter to regulate τ_{ij}. n_{ij} is the closeness factor of the city i from city j and set to $1/d_{ij}$, where d_{ij} is the distance between city i and j. β is the parameter to regulate n_{ij}. U_k is the set of unvisited cities for each ant k.
By intuition, starting with l ants random distributed to each city. After n iterations (n is the number of cities), each ant has completed a tour. The shorter tour would have a higher chance of being traveled by more ants. We need a set of equations to model the pheromone level of each trail between cities as they are being traversed by each ant k.

$$\tau_{ij}(t+1) = \rho \cdot \tau_{ij}(t) + \Delta\tau_{ij} \tag{2}$$

$$\Delta\tau_{ij} = \sum_{k=1}^{l} \Delta\tau_{ij}^{k} \tag{3}$$

$$\Delta\tau_{ij}^{k} = \begin{cases} Q/L_k & \text{if ant } k \text{ travels on edge } (i,j) \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

Looking at equation 4, the function Q/L_k represents the increase in pheromone level, where Q is a constant and L_k is the length of the tour by ant k from city i and j in one iteration. If no ants traveled the path, the change is zero. Equation 3 represents the total increase of pheromone for a path after taking account of all ants travel through the path after one iteration. Equation 2 is the update function, $\rho \in [0,1]$ is the parameter for regulating the reduction of τ_{ij}. Given the sets of above equations, Let's take a look at the Pseudocode of Ant Colony Algorithm for solving TSP.

Pseudocode

Let $U_k = \{x | x \in X \text{ and } x \notin G, \exists G \in tabu_k\}$

1. Initialize
   Set time=0
   For every edge(i,j) set an intial $\tau_{ij} = c$ for trail density and $\delta_{ij} = 0$

2. Set s=0
   For k = 1 to l
   Place ant k on a city randomly. Placed city in $visited_k$.
   Place the group of city in $tabu_k$

3. Repeat until s $\leq$m
   Set s = s + 1
   For k = 1 to l
   Choose the next city to be visited according to the p_{ij}^k by equation 1
   Move the ant k to the selected city
   Insert the selected city in $visited_k$
   Insert the group of selected city in $tabu_k$

4. For k = 1 to l
   Move the ant k from $visited_k(n)$ to $visited_k(l)$
   Compute the tour length L_k traveled by ant k
   Update the shorest tour found
   For every edge (i,j)
   For k = 1 to l
   Update τ_{ij} according to equation 2 to 4 time++

5. If (time ¡ timeMax)
   Empty all $visited_k$ and $tabu_k$
   Goto Step 2
   Else
   Print the shortest tour
   Stop