Daniel Jarc (jarcd)
CS325-400: Analysis of Algorithms
May 27, 2018

Project Research

Traveling Salesman Problem (TSP)

You are given a set of n cities and for each pair of cities c_1 and c_2, the distances between them $d(c_1, c_2)$. Your goal is to find an ordering (called a tour) of the cities so that the distance you travel is minimized. The distance you travel is the sum of the distances from the first city in the ordering to the second city, plus the distance second city to the third city, and so on until you reach the last city, and then adding the distance from the last city to the first city. For example, if the cities are Seattle, Portland, Corvallis and Boise. The total distance traveled visiting the cities in this order is:

d(tour) = d(Seattle, Portland) + d(Portland, Corvallis) + d(Corvallis, Boise) + d(Boise, Seattle)

In this project, you will only need to consider the special case where the cities are locations in a 2D grid (given by their x and y coordinates) and the distance between two cities $c_1 = (x_1, y_1)$ and $c_2 = (x_2, y_2)$ is given by their Euclidean distance. To avoid floating point precision problems in computing the square root, we will always round the distance to the nearest integer. In other words, you will compute the distance between cities c_1 and c_2 as:

$$\mathrm{d}(c_1, c_2) = \text{nearestint}\left(\sqrt{((x_1 - x_2)^2 + (y_1 - y_2)^2}\right)$$

For example, if the three cities are given by the coordinates $c_1 = (0, 0)$, $c_2 = (1, 3)$, $c_3 = (6, 0)$, then a tour that visits the cities in order $c_1 \rightarrow c_2 \rightarrow c_3 \rightarrow c_1$ has the distance:

$$\mathrm{d}(tour) = \mathrm{d}(c_1, c_2) + \mathrm{d}(c_1, c_2) + \mathrm{d}(c_1, c_2)$$

where:

$$\mathrm{d}(c_1, c_2) = \text{nearestint}\left(\sqrt{((0-1)^2 + (0-3)^2}\right)$$
$$\mathrm{d}(c_1, c_2) = \text{nearestint}\left(\sqrt{(-1)^2 + (-3)^2}\right)$$
$$\mathrm{d}(c_1, c_2) = \text{nearestint}\left(\sqrt{1+9}\right)$$
$$\mathrm{d}(c_1, c_2) = \text{nearestint}\left(\sqrt{10}\right)$$
$$\mathrm{d}(c_1, c_2) = \text{nearestint}(3.16)$$
$$\mathrm{d}(c_1, c_2) = 3$$

$$\mathrm{d}(c_2, c_3) = \text{nearestint}\left(\sqrt{((1-6)^2 + (3-0)^2}\right)$$
$$\mathrm{d}(c_1, c_3) = \text{nearestint}\left(\sqrt{(-5)^2 + (3)^2}\right)$$
$$\mathrm{d}(c_1, c_3) = \text{nearestint}\left(\sqrt{25+9}\right)$$
$$\mathrm{d}(c_1, c_3) = \text{nearestint}\left(\sqrt{34}\right)$$
$$\mathrm{d}(c_1, c_3) = \text{nearestint}(5.83)$$
$$\mathrm{d}(c_1, c_3) = 6$$

$$\mathrm{d}(c_3, c_1) = \text{nearestint}\left(\sqrt{((6-0)^2 + (0-0)^2}\right)$$
$$\mathrm{d}(c_3, c_1) = \text{nearestint}\left(\sqrt{(6)^2 + (0)^2}\right)$$
$$\mathrm{d}(c_3, c_1) = \text{nearestint}\left(\sqrt{36+0}\right)$$
$$\mathrm{d}(c_3, c_1) = \text{nearestint}\left(\sqrt{36}\right)$$
$$\mathrm{d}(c_3, c_1) = \text{nearestint}(6)$$
$$\mathrm{d}(c_3, c_1) = 6$$

$$\mathrm{d}(tour) = 3 + 6 + 6 = 15$$

Ant Colony Optimization (ACO)

The Ant Colony Optimization algorithm was proposed by Marco Dorigo in 1991 and published in 1992. [1] ACO was inspired by prior research conducted by Goss, Aron, Deneubourg and Pasteel on the behavior of Argentinian ants. [2]

Ants have limited individual capabilities but can accomplish complex tasks using pheromones to modify their environment, eliminating the need for direct communication and allowing complex group behavior and problem solving. A common example used is foraging for food. The goal is to gather the most food while expending the least amount of energy, leading to an optimized solution. As ants move back and forth between the food source and the nest, pheromones are deposited on the ground. The shortest path will be traveled more often because it takes the least amount of time and is traveled more often, resulting in higher levels of pheromones and an increased probability of ants choosing that path in the future. Alternative, longer and less efficient paths will be traveled less often, resulting in pheromones evaporating and a decreased probability of ants choosing those paths. The concept of a trail becoming increasingly attractive as it is traveled more often is considered autocatalytic behavior – *a positive feedback loop*.

The Double Bridge experiments [2] studied the behavior of ants given different configurations of a bridge structure and a food source. Reference the diagrams below. The first configuration (1) was a split bridge with two equal length paths. Initially the paths were chosen with equal probability but over time, one path eventually accrued more pheromones, leading to a preferred path. In another version of the experiment (2), a double bridge with one long path and one short path was presented. The shorter path rapidly accrued pheromones due to the reduced time it took ants to travel and became the preferred path. In a third experiment (3), one long path was presented, and a shorter path was introduced later. The long path remained the preferred path due to the established pheromone trail; resulting in a missed opportunity - this problem is addressed in the artificial model. All three of these experiments demonstrate the concept of pheromones, probability and a positive feedback loop.

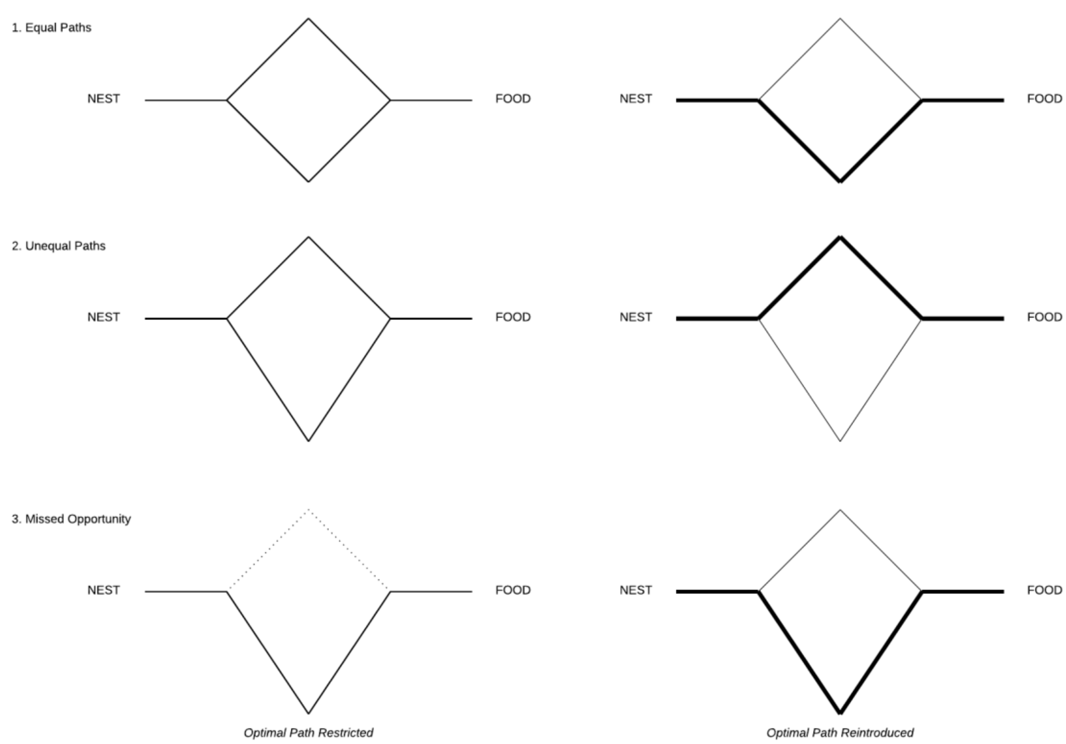

Figure 1-1: Double Bridge Experiment

Artificial ants are designed to mimic many aspects of their natural counterparts, with some additional capabilities to avoid "missed opportunities" and loops while traversing a graph structure. The artificial agents operate in two main working modes:

1. *Forward – Probabilistic Search*
    a. Choose nodes/vertices based on pheromone trail.
    b. Do not deposit pheromones while moving.
2. *Backward – Solution Construction*
    a. Prior to starting back to nest, eliminate loops.
    b. Retrace steps, deposit pheromones on "solution" path.

ACO to solve TSP

The Ant Colony Optimization was initially developed to solve the Traveling Salesman Problem. If there are *n* cities, then *l* distinct ants start from any of these *n* cities, randomly. The ants have the following additional characteristics:

- They will not visit a city more than once.
- They know the distance of the cities and tend to choose the nearby city (if all other factors are the same).
- If the distance of two paths is the same they tense to choose path with more pheromones.

The probability P_{ij}^k to select city j by ant k, sitting at city i is given by the following formula. If a city is not in the allowed set, then there is a 0 probability of ant k choosing that city.

$$P_{ij}^k = \begin{cases} \dfrac{[\tau_{ij}]^\alpha \cdot [\eta_{ij}]^\beta}{\sum_{\delta \in allowed_k} [\tau_{i\delta}]^\alpha \cdot [\eta_{i\delta}]^\beta} & j \in allowed_k \\ 0 & otherwise \end{cases}$$

$$\tau_{ij} = Intensity\ of\ pheromone\ between\ city\ i\ and\ j$$

$$\alpha = Influence\ of\ pheromones\ on\ probability, constant, \alpha \geq 0$$

$$\eta_{ij} = Visibility\ of\ city\ j\ from\ city\ i = \frac{1}{d_{ij}}$$

$$\beta = Influence\ of\ visibility\ on\ probability, cosntant, \beta \geq 1$$

After each ant completes a complete tour, the pheromones are updated in such a way that shorter paths are given more pheromones than longer paths:

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

$$\Delta\tau_{ij} = \sum_k^l \Delta{\tau_{ij}}^k$$

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

$$\mathrm{t} = \text{Iteration counter, time}$$

$$\rho = \text{Pheromone evaporation factor, constant between 0 and 1}$$

$$\Delta\tau_{ij} = \text{Total increase of pheromone level on edge (i, j) by all ants}$$

$$L_k = \text{Total distance of tour by ant k}$$

$$Q = \text{constant}$$

Putting it all together, using ACO to solve TSP can be accomplished as follows:

1. Initialize all edges in graph *G* with a uniform pheromone level.
2. Randomly place *l* ants on the graph.
3. For each ant, probabilistically select a path.
4. Globally update pheromone levels on graph edges. Repeat until iteration limit is reached.

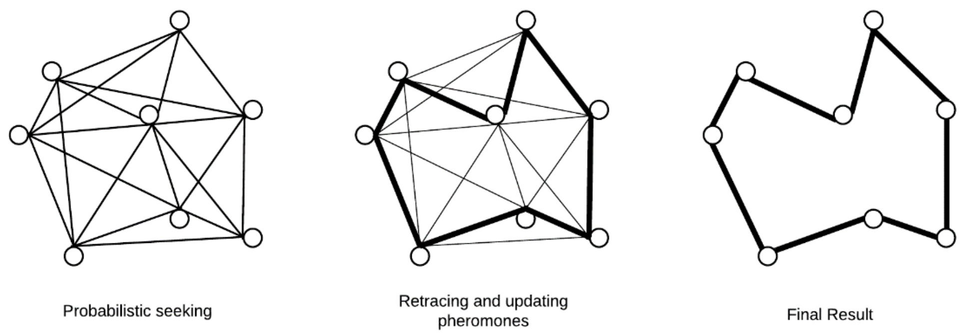

Figure 1-2: ACO solving TSP

Pseudo-code

```
for t=1 to iteration-threshold                      // For desired iterations
        for k=1 to l ant                            // For each ant
                for move_count=1 to n       // Visit all cities
                        move ant based on P_ij^k  // Probability formula
        calculate L_k                               // Ant's total distance traveled
update pheromone Δt_ij                      // Update pheromones
```

References

[1] M. Dorigo, *Optimization, Learning and Natural Algorithms*, PhD thesis, Politecnico di Milano, Italy, 1992

[2] S. Goss, S. Aron, J.-L. Deneubourg et J.-M. Pasteels, *Self-organized shortcuts in the Argentine ant*, Naturwissenschaften, volume 76, pages 579-581, 1989