Artificial Intelligence Lab Assignment 4

Group Number: 27

Pavan Kumar V Patil 200030041

Karthik J Ponarkar 200010022

February 6, 2022

Contents

1 Problem Description

Given a set of cities (coordinates) and distances between them, we need to find the best (shortest) tour (visiting all cities exactly once and returning to the origin city) in a given amount of time, viz. Traveling Salesman Problem.

1.1 Requisite inputs for the program

- First line in the input file denotes whether the graph is Euclidian or non-Euclidian.
- Next line contains the number of cities (vertices) N.
- Next N lines contains the coordinates of each city. i.e. $(x_i, y_i)\ \forall\ i \in [1, N]$
- Further, next N lines contains the distance of every other city from the current city, i.e. $d\,[i]\,[j]$ where i is the current city and $i, j \in [1, N]$. Pictorially it represents a 2-D matrix of edges of size $N \times N$

2 Overview of the approach to the solution

- The problem asks us to optimize the tour conducted by the salesman and it is does not constraints us to choose any particular city to start the tour.
- For driving our main algorithm towards finding an optimized least cost path, we conduct greedy tours for all the cities considering each as the starting city. We choose the city with best tour as the start city in the Ant Colony Optimization algorithm and deposit some additional pheromone on that tour path.
- ACO algorithm is a probabilistic technique for solving computational problems which can be reduced to finding good paths through graphs.
- Several artificial ants are made to find the optimal solution. In the first step of solving a problem, each ant generates a tour. In the second step, tours found by different ants are compared based on the tour cost. And in the third step, best tour is retained and pheromone for all the edges connecting the cities are updated.
- This is process is iterated until we approach an optimal solution to the problem.
- Detailed algorithm with pseudocode will be discussed in further sections.

3 ACO algorithm and analysis

- Requisite Variables:
  - N - Number of Cities
  - M - Number of Ants
  - ρ - Evaporation rate of pheromone ($0 \leq \rho \leq 1$)
  - Q - A constant introduced for calculating visibility factor
  - α - A constant used for tuning the effect of pheromone factor on probability.
  - β - A constant used for tuning the effect of visibility factor on probability.
- We create a 2-D array of pheromone deposited on each possible edge connecting any two cities. The net pheromone deposited on each edge after we conduct the tours for M ants is equal to the arithmetic sum of old pheromone left after evaporation and new amount of pheromone deposited by the M ants.
- $\tau_{ij}(t)$ represents the amount of pheromone on the segment from the i^{th} city to j^{th} city at time t.
$$\tau_{ij}(t+n) = (1-\rho)\,\tau_{ij} + \Delta\tau_{ij}(t, t+n)$$
- The amount of pheromone deposited by the k^{th} ant is given by
$$\Delta\tau_{ij}^{k}(t, t+n) = Q/P_k$$
where P_k is equal to the tour cost conducted by the k^{th} ant.
- During the inceptive phase, we deposit some fixed amount of pheromone Δ_{o} on each edge (i, j), where $(i, j) \in E(G)$
- Each ant constructs its tour in greedy probabilistic manner. The probability that the ant chooses the j^{th} city being present at the i^{th} city depends upon the pheromone deposited on the edge connecting these two cities and the visibility factor of that edge.
- We introduce some quantities in order to define the probability for the ant.
$$p_{ij}^{k} = \frac{\tau_{ij}^{\alpha}(t) \cdot \chi_{ij}^{\beta}}{\sum_{m \in \mathbf{A}} \tau_{im}^{\alpha}(t) \cdot \chi_{im}^{\beta}}$$
where A is the set of neighbouring cities that have a direct edge from i^{th} city and χ_{ij} is the visibility factor for the ant to choose that edge.
$$\chi_{ij} = \frac{Q}{P_k}$$

4 Pseudocode

```
function TSP-ACO(start, ρ, Q, α, β)
    global N
    global pheromoneDepo
    global cities
    bestTour = NULL
    while some termination criterion is met do
        tourList = [ ]
        for each ant in M ants do
            closedList = [ ]
            current = start
            while True do
                if closedList.length() == N − 1 then
                    break
                end if
                bestValidCity = selectBest(pheromoneDepo, current)
                bestValidCity.parent = current
                current = bestValidCity
            end while
            tourList.append(currentTour)
        end for
        updateRemainingPheromone(pheromoneDepo, ρ)
        for each tour in tourList do
            for each edge in tour do
                edge = (i, j)
                pheromone[i, j] = pheromone[i, j] + Q/P_k
            end for
        end for
        bestLocalTour = selectBestTour(tourList)
        if bestLocalTour.cost() ≤ bestTour.cost() then
            bestTour = bestLocalTour
        end if
    end while
    return bestTour
end function
```

5 Results

- Minimal tour cost obtained by the ACO algorithm for the Euclidian graph and $N = 100$ is 1634
- Minimal tour cost obtained by the ACO algorithm for the non-Euclidian graph and $N = 100$ is 5331.
- This result varies during every execution of the program because ACO follows a probabilistic greedy algorithm.