Practical 3: Ant Algorithm for Solving TSP

Yushuo Chen
2020101918
MAM

2023 年 11 月 20 日

Introduction

In this assessment report, I am going to solve challenge B which aims to solve TSP problem using ant algorithm. [1]

1 Problem Description

TSP problem is in order to find a minimal length closed tour that visits each city once. My report will reveal the path which obtain the optimal distance based on dataset *st70* from homework 2. And I will compare algorithm AA (ant algorithm) and GA (genetic algorithm) solving the same TSP problem.

2 Solution Encoding

Since we need to obtain a cyclic path of 70 cities, we can represent one path as a permutation **e.g.** (2 3 7 6 1 5 4), where the number 1 represents city A and number 2 represents city B etc. Let t represent a solution as this permutation, and $\mathcal{T}$ be the set of all permutations of cities set i.e.the search space.

Then we can calculate the Euclidean distance between each city according the locations.

$$D(i(x_1, y_1),\ j(x_2, y_2)) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}.$$

[1] All codes attached to homework could be found in `https://github.com/CyanCap/CI-code.git`.

So the total distance during a tourment is:

$$dis = [\sum_{i>=1}^{7} D(t(i), t(i+1))] + D(t(1), t(8)).$$

3 Objective Function And Constraint

Obviously, we want to find the minimal of the distance dis, objective function is:

$$t^* = arg\min_{t \in \mathcal{T}} \ dis(t).$$

Moreover, there will be just a few of constraints which I can illustrate straightly. Firstly, each solution i.e.permutation must contain all cities. Secondly, each solution must contain a specific city only once. So a solution is a vector of the size $1 \times n$, n is the number of cities.

4 Models for Ant-cycle AS

In solving this problem, I decide to use model ant-cycle ant system. AS is developed by ACO (ant colony optimization). It simulates the swarm of ants finding foods according the pheromones they leave. Each ant's decision is based on a probability

$$p_{ij}^k(t) = \begin{cases} \dfrac{\tau_{ij}^{\alpha}(t)\eta_{ij}^{\beta}(t)}{\sum_{u \in \mathcal{N}_i^k(t)} \tau_{iu}^{\alpha}(t)\eta_{iu}^{\beta}(t)}, & \text{if } \ j \in \mathcal{N}_i^k(t) \\ 0, & \text{if } \ j \notin \mathcal{N}_i^k(t) \end{cases}$$

where $\tau_{ij}(t)$ represents the residual amount of pheromones from node i to node j i.e.pheromone intensity. And $\eta_{ij}(t)$ is the factor that represents heuristic information. Then we should choose the way updating the pheromone as time t goes by.

Firstly, we can generate the update function:

$$\tau_{ij}(t+1) = (1-\rho) \cdot \tau_{ij}(t) + \sum_{k=1}^{n_k} \Delta\tau_{ij}^k(t)$$

Secondly, since we not only need to consider the correct direction of each choice, but also need to think of the minimal distance from node to node and the information an ant walks so far. We use **ant-cycle AS** to represent the way pheromone deposits i.e.$\Delta\tau_{ij}^k = Q/L_k$.

5 Algorithm Design With Pseudo Code And Parameters

Here is my pseudo code:

Algorithm 1: Ant-cycle Ant System

Input: α, β, ρ, Q, n_k, τ_0, η

Data: locations information from *st80.csv*

Result: optimal tourment t^*

```
while stopping condition is false do
    place n_k ants randomly;
    construct a path for n_k ants;
    compute dis_k;
    reserve the best tourment;
    evaporate and update pheromone;
    reset all ants;
end
```

Now we should set all parameters which must fit our algorithm and problem better.

- Population of ants: $n_k = 100$. Population with 100 ants may perform well since amounts of ants must greater than that of destinations in nature.
- Decision process biases: $\alpha = 2$, $\beta = 5$. I want the ants to be more likely to wander randomly (τ^α) and be more creative (η^β).
- Factor of pheromone evaporate: $\rho = 0.2$. Pheromones should last long because that we want reserve some important informations after setting α and β.
- Factor indicating power of pheromone: $Q = 100$.

6 Result And Analysis

I obtain the final result is

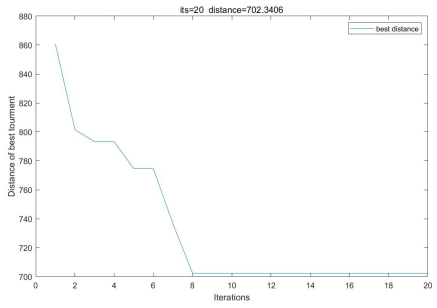
$$dis_{\min} = 702.3406$$

with tourment in variable *path_best*.

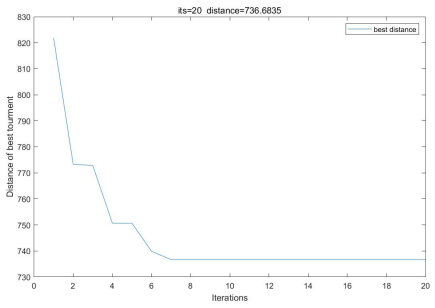
And I found that this result appeared after iteration 8, which means this algorithm perform very quickly. Moreover, the line sees a rapid decline when it begins, and then

reaches a minimum and stays in that level. After operating such a lot of experiments, interest phenomenons happen: algorithm usually reaches minimum but local minimum in the most of times.

We can see a dramatical decrease at first, then a straight line parallel to x-axis. As you can see, 2 figures share the same shape and similar result. We cannot conclude more unless AS executes quickly and easy hitting (local) minimum.

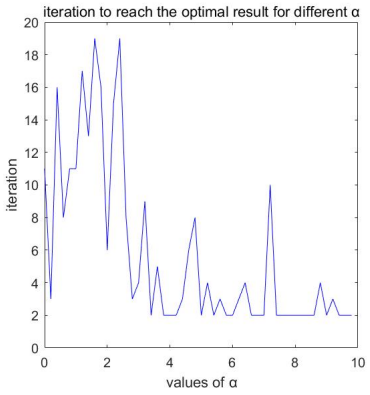


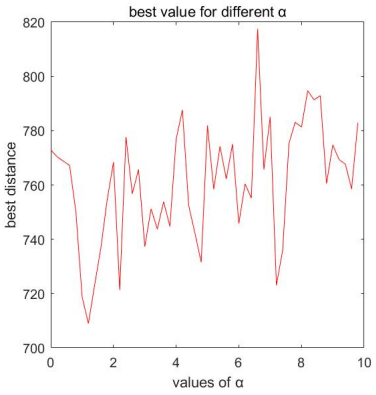
(a) experiment-1



(b) experiment-2

7 More Discussions





Moreover, if we analyze effects about different α, we can find that α needn't to be too large or small. The left graph almost shows a convex "∩" shape. While the right graph has showed a increasing trade.

Now we can compare a little bit of AS and GA. In homework 2, I've designed genetic algorithm to solve same TSP problem with same dataset. But GA has cost much more time than AS. This difference almost could be 1000 times of cost. And the final result is even worse than using AS.