3510 TSP Project

Alex Elliott and Brenden Waits

April 20, 2020

1 Introduction

Here are quick facts about our algorithm:

- It uses simulated annealing, with a cooling rate of .001% and a starting temperature of 1000.
- It pre-computes all distances between provided nodes into a 2D array at the beginning.

2 Pseudocode

Here is the pseudocode for our algorithm. This code illustrates how the $sa()$ function works in our submission. The rest of the code is for outputting our solutions to the specified file.

```
Let t = 1000
Let cr = .00001
Let nodes = list of nodes from passed in
current = nodes
best = nodes
while $t > 1$ do
    new = current
    swap nodes at two random indexes in new
    cd, nd = tour distances of current and new, respectively
    if (exp((cd-nd)/t)) > random float between 0 and 1 then
        current = new
    end
    if new tour distance is less than current then
        best = new
    end
    t *= 1 - cr
end
return best
```

3 English

The algorithm is pretty straight forward. The idea behind it is nodes in the tour are randomly swapped, and if the new tour distance is less that the old one the best tour is updated. This would theoretically run forever, which is where the simulated annealing concept comes in. As the algorithm runs, the starting temperature decreases at a specific rate. Here, we defined the initial temperature as 1000 and the cooling rate as .00001, which means that each iteration the temperature decreases by .001%. These values were selected by trail an error. There is another degree of randomness to this. The current list is not always updated with the swapped list. Using the formula $e^{\frac{cd-nd}{t}}$, the new distance is subtracted from the current distance and divided by the temperature. This number gets larger and larger as the temperature gets smaller and smaller. The result of this operation after being checked against the random float is current gets swapped with new more often as the temperature is higher. As the temperature decreases, the algorithm cools, which means less random jumps and changes. Once the temperature reaches 1 the algorithm stops and a optimal solution is found. This process is done 10 times, the lowest found is the one that is finally returned by our program.