Introduction to Intelligent Systems - Lab 3

Diego Velasco Volkmann Eduardo Faccin Vernier
S2851059 S3012875

October 31, 2015

Assignment

For the program to meet the assignment specifications some changes in the source code were made. The first change regards the temperature value that must remain constant throughout the algorithm execution. To accomplish that, the line with the content ”temp = temp*0.998;” in the file ”tspinit.m” was commented out. This way, the 0,2% temperature decrease that would happen every 100 iterations doesn’t occur.
To assure the algorithm performs (at least) 100x100 single steps, we set ”maxsteps” to 300 in every function call (i.e. tspinit(50, 300, 0.01, 1)).
To calculate the mean value <l>and the variance of the set over the last 50 measured values, the last 50 results were kept in an array, and at the end of the execution, it’s mean and variance were calculated. These values were then returned to the caller function and the result for every T value was plotted using MATLAB’s errorbar function.

About T-dependence: The T value is responsible for allowing locally bad decisions. That may sound counterproductive, but in a space with many local minima, accepting worse solutions allows for a more extensive search for the optimal solution. Greater T values increase the chances of locally bad choices occurring, which in a global sense may help explore the solution space in a more thorough way. Smaller T values will mostly accept solutions that decrease the total size of the current path, but occasionally may accept a worse solution. A T value equal to zero would make the algorithm greedy, accepting only solutions that would decrease the path length value, this will lead to a quick conversion to a local minima, which wouldnt necessarily correspond to the global solution.
The Simulated Annealing technique makes use of the Metropolis algorithm starting with a high T value, allowing trades that increase the total path length, in order to access a large part of the solution space. The T value is decreased throughout the execution of the algorithm, which leads to a local minima that is hopefully closer to the optimal solution.

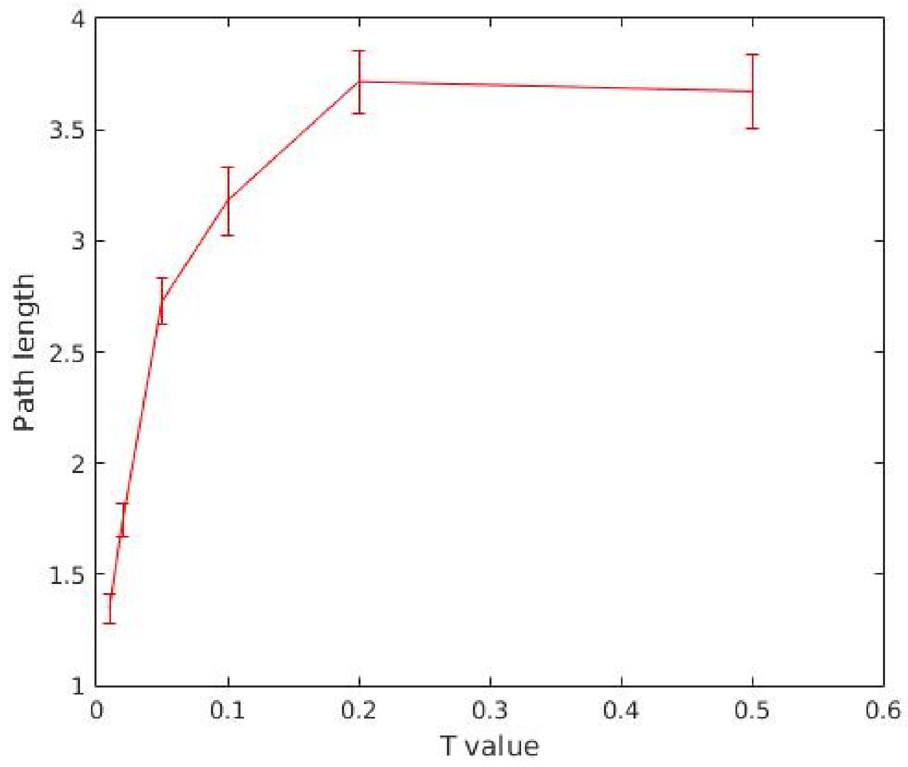

Figure 1: Plot for T values of 0.5, 0.2, 0.1, 0.05, 0.02, 0.01 with the number of cities equal to 50 and 30000 iterations of the algorithm for each T value

The data in the plot shows that higher temperatures, considering Metropolis algorithm with no temperature decrease, correspond to longer paths. Higher temperatures increase the probability of choosing ”bad paths”, and as the temperature does not drop during execution, these bad decisions keep being made in the same rate throughout the algorithm run time, therefore it might not even converge to a local minima in the end of the execution. With low temperatures, as in T = 0.01, for example, the number of bad decisions is much lower, this restrict the exploration of the solution space, which might ultimately trap it in a local minima that isn’t remotely close to the global minima. A hybrid approach, starting with a high temperature that is decreased throughout the algorithm execution would allow a great solution space exploration and would lead to a good local minima conversion.

Division of work

The plot and code change was done by Eduardo, while Diego researched and wrote about T-dependence whilst reviewing the result. Later Eduardo decided to change Diego’s answers.

Appendix: Plotter function source code

Listing 1: plotter.m file

```
xAxis = [0.5 0.2 0.1 0.05 0.02 0.01];

index = 1;
for tValue = xAxis
    temp = tspinit(50, 300, tValue, 1);
    plotdata(index,1) = temp(1);
    plotdata(index,2) = temp(2);
    index = index + 1;
end

figure
errorbar(xAxis, plotdata(:,1), plotdata(:,2), 'r')
ylabel('Path length');
xlabel('T value');
```