Alexandria University

Faculty Of Engineering

Computer and Systems Engineering Department

Applying Simulated Annealing to Traveling Salesman Problem

Authors:
Aya Aboud [1]
Ramy Wagdy [20]
Mina Makram [70]

Supervisors:
Prof. Dr. Saleh ElShehaby
Eng. Alaa

May 16, 2016

ALEXANDRIA UNIVERSITY

Abstract

1 Introduction

1.1 Traveling Salesman Problem Definition

The traveling salesman problem is a problem where given a set of cities and the distance between each 2 cities, it is required to find the shortest distance tour passing through each city exactly once and returning to the starting city. This problem is represented with a graph where each city is represented as a node and the distance as the edge cost between nodes. In this graph it's required to find the minimum-weight Hamilton circuit.

1.2 Motivation

TSP is an NP-hard (and also NP-complete) problem. This means that there is no algorithm that can find the minimal tour in polynomial time. Using Brute-force algorithm would take running time of O(N!). This could be accepted for small N, but for large N (greater than 25) this would take hours and years. The running time for different values of N are shown in table 1 assuming 10^{21} steps per second.

Consequently, we need to find an algorithm that can run in polynomial time to overcome the large running time of classical algorithms. This algorithms are called **heuristic algorithms**. In the following sections we discuss heuristic algorithms and how it is implemented.

Table 1: Running time of Brute-force for different N

Number of nodes	**N!**
20	2.4 milliseconds
25	4.2 hours
30	8,400 years
35	326 billion years

A heuristic is a technique designed for solving a problem more quickly when classic methods are too slow

2 Simulated Annealing

2.1 Background

Annealing is the process of heating up a solid and then cooling it slowly until it crystallizes. The atoms have high-energy values at very high temperatures. This gives the atoms a great freedom in their ability to restructure themselves. As the temperature is reduced, the energy levels of the atoms decrease. If the cooling process is carried out too quickly, defects will occur in the crystal structure. The temperature should be reduced slowly to allow a more consistent and stable crystal structure to form. Simulated annealing Algorithm tries to follow this process. It begins at a very high temperature, at which it can explore a wide range of search space. As the process proceeds, the temperature is allowed to fall, thus restricting the search space. This often leads the simulated annealing algorithm to a better solution, just as a metal achieves a better crystal structure through the actual annealing process.

2.2 The Algorithm

Initially, Simulated Annealing starts with a random initial solution and sufficiently large temperature t_0. The algorithm tries to find the global optimal solution. Simulated Annealing goes as follows:

- First, it modifies the solution using some technique and then recomputes the Cost of the solution with some Cost Function.
- Secondly, it compares the new solution with the previous solution. if the new solution's cost is less than the previous solution, the new solution is accepted. Otherwise, the new solution is accepted with probability $e^{-\frac{\Delta}{T}}$ where,
  - Δ: is the difference between the new computed solution the (called working solution) and the current solution;
  - T: is the current temperature.
- Thirdly, After a fixed number of iterations, the temperature is decreased by the cooling rate α, such that $T_N = \alpha * T_{N-1}$,
  where $0 < \alpha < 1$.

Algorithm 1 Simulated Annealing Algorithm

```
 1: procedure SimulatedAnnealing()
 2:     Get an initial solution $S$
 3:     Get an initial temperature $T \geq 0$
 4:     while $T > 0$ do
 5:         for $i \leftarrow 1$ to L do
 6:             Pick a random neighbour S' of S
 7:             let $\Delta = cost(S') - cost(S)$
 8:             if $\Delta < 0$ (downhill move) then
 9:                 Set S = S'
10:             end if
11:             if $\Delta > 0$ (Uphill move) then
12:                 Set S = S' with probability $e^{(-\Delta/T)}$
13:             end if
14:         end for
15:         Set $T = \alpha T$ (reduce temperature)
16:     end while
17: end procedure
```

The Algorithm stops when the temperature reaches a value very close to zero. Figure 2.5 shows the general algorithm of Simulated Annealing.

2.3 Simulated Annealing in TSP

In the previous section, SA was discussed in an abstract way although the general concepts applies to TSP. In this section, solution representation and modifying the solution are discussed in details.

2.3.1 Solution Representation

Cities are represented in an array where traveling through the cites is by visiting the cells of the array in order. Each City is represented by

2.3.2 Cost Function

Cost function is the sum of the Euclidean Distances between consecutive cities in the array.

2.3.3 Solution Modification

Solution is modified by swapping 2 cities in the array. The 2 cities are chosen randomly.

2.4 Why Simulated Annealing

Accepting a worse solution is what makes Simulated Annealing find global optimal solution and not get stuck in a local optimal one. Moreover, we notice that at high temperature, the probability of accepting a worse solution is large according to $e^{-\frac{\Delta}{T}}$. This coincides with the general idea of annealing where at large temperature, the molecules can move freely. Similarly, at high temperature, the algorithm accepts worse solution with high probability which gives it the opportunity to explore a large part of the search space. As the temperature decreases, the probability also decreases to maintain the good solution obtained so far.

3 Challenges

Parameters (initial temperature, iterations and alpha) values affect the cost and time.

- If the initial Temperature is too low, then worse solution will barely be selected making it get stuck in the local minimum. If the initial Temperature is too high, it would make the solution accept worse solutions with high probability leading to skipping good solutions.
- The cooling rate α should be slow (≥ 0.95).

4 Implementation

4.1 Data sets

We have designed our data sets such that 1 meter only between each two cities. The cities are being in square shape. So the optimal cost must be equal to the number of the cities.

4.2 Parameter Tuning

We applied parameter Tuning on Initial Temperature, Iterations per Temperature and Cooling rate to get the parameter values for resulting in the best solution. Table 2 shows the parameters with their corresponding values and their effect on running time and Cost (Optimal Path). As shown, the best parameters which give the best cost are colored in red.

Table 2: Parameter Tuning with Time and Cost

Alpha	**Initial Temperature**	**Iterations**	**Running Time**	**Cost**
0.99	5	50	1 min 0 second	2160.849
0.99	5	100	1 min 54 seconds	2010.317
0.99	5	200	3 min 58 seconds	1259.154
0.99	10	50	1 min 10 seconds	1724.638
0.99	10	100	2 min 18 seconds	1656.86
0.99	10	200	4 min 31 seconds	1163.086
0.99	30	50	1 min 24 seconds	2214.038
0.99	30	100	2 min 50 seconds	2166.432
0.99	30	200	5 min 46 seconds	1993.711
0.99	50	50	1 min 31 seconds	2918.765
0.99	50	100	2 min 49 seconds	2547.659
0.99	50	200	3 min 47 seconds	2071.2608
0.99	100	50	1 min 10 seconds	3211.1058
0.99	100	100	2 min 2 seconds	2555.7038
0.99	100	200	4 min 6 seconds	2040.8708
0.95	5	50	0 min 7 seconds	3093.5708
0.95	5	100	0 min 14 seconds	2552.9162
0.95	5	200	0 min 29 seconds	2228.2673
0.95	10	50	0 min 8 seconds	2830.9144
0.95	10	100	0 min 16 seconds	2022.1137
0.95	10	200	0 min 32 seconds	1877.3376
0.95	30	50	0 min 10 seconds	3004.9478
0.95	30	100	0 min 19 seconds	2455.3720
0.95	30	200	0 min 38 seconds	2354.4494
0.95	50	50	0 min 10 seconds	3667.2455
0.95	50	100	0 min 19 seconds	3739.9705
0.95	50	200	0 min 40 seconds	3326.5577
0.95	100	50	0 min 12 seconds	5146.5861
0.95	100	100	0 min 22 seconds	4184.6929
0.95	100	200	0 min 40 seconds	3369.7211

5 Results

- 50 city dataset:
  time: 3 min 38 seconds cost: 50
- 100 city dataset:
  time: 4 min 16 seconds cost: 169.657
- 200 city dataset:
  time: 4 min 3 seconds cost: 653.79

6 Conclusion

NP-hard Problems are impossible to solve specially when the problem size is relatively large. Heuristic Algorithms can be used to reach (sub)optimal solution. Simulated Annealing is one of the heuristic algorithms that can be used to solve TSP. In relatively small cities, Simulated Annealing nearly reach the optimal solution. However, in large number of cities, SA reaches a sub-optimal solution.

7 References

1. https://www.math.ku.edu/ jmartin/courses/math105-F11/Lectures/chapter6-part3.pdf
2. E Aycan and T Ayav. Solving the course scheduling problem using simulated annealing. In Advance Computing Conference, 2009. IACC 2009. IEEE International, pages 462–466. IEEE, 2009.