Traveling Salesman Problem with Simulated Annealing

Meifang Li
13043390
meifang.li@student.uva.nl
17/12/2020

Yuhao Qian
13011456
yuhao.qian@student.uva.nl
17/12/2020

ABSTRACT

The Traveling salesman problem is a famous NP-hard problem and could be solved in several ways. We implemented the Simulated Annealing algorithm trying to find the shortest distance for 280 cities and 442 cities. After choosing a suitable cooling schedule and investigating the effect of initial temperature, number of cooling steps, cooling ratio and Markov chain's length, we found the shortest distance for 280 cities was 2962.48 ± 59.51 and for 442 cities was 57969.0144.

1 INTRODUCTION

The traveling salesman problem aims to find the shortest way to visit all the cities exactly once and return to the starting point. If the order of n cities is taken as a permutation and the clockwise and anti-clockwise tours are taken as the same order, then the number of total possible solutions is $\frac{(n-1)!}{2}$. When n is small, it is easy to compute all the distances of these solutions and get the minimal one. However, this enumeration method would cost extremely large computational power when n becomes large. Thus various algorithms could help to achieve the best solution, one of which is simulated annealing.

Simulated annealing (SA) is a probabilistic method to find the global minimum of a cost function that may contain local minimums[1]. In the cooling process from a high temperature to a low temperature, the traveling tour would be perturbed by reversing all the positions between two randomly chosen cities in the permutation, then the total distance would be compared with the previous tour and accepted or rejected depending on the current temperature. A larger cost could also be accepted with certain probabilities at a high temperature, resulting in more searching space and a better chance for finding a global minimum. Thus SA could converge into the minima as the temperature goes down.

In this report, we aimed to figure out the shortest route for the set of 280 cities so we explored the parameters that could impact the final solutions. First, three different cooling schedules were compared: linear, exponential, quadratic. Then the initial temperature, the number of cooling steps, and cooling ratio were discussed. The effect of Markov chain's length was evaluated at last and a final solution could be obtained depending on these parameters. The result may be a local minimum due to limited computing power but showed great convergence in correspondence with our expectations. We also tried to address a bigger problem with the set of 442 cities and searched for the best solution we could get.

2 THEORY

2.1 Traveling Salesman Problem

The traveling salesman problem (TSP) can be viewed as one of the best-known problems in computing fields. Given a certain number of cities along with the distances between each pair of them, a traveling salesman would depart from one city, go through all other cities only once and get back to the starting point. The TSP is to find the shortest way to visit all the cities, in which the order of these cities should be optimized. An illustration of the tour is shown in Figure 1.

Figure 1: TSP example: Shortest route that visits the 15 largest cities in Germany[2].

The order of the cities could be viewed as a permutation. Let P_n be the collection of all permutations of the set $\{1, 2, \cdots, n\}$ and c_{ij} denote the distance between node i and node j. Then the TSP is to find a $\pi = (\pi(1), \pi(2), \cdots, \pi(n))$ in P_n such that $c_{\pi(n)\pi(1)} + \sum_{i=1}^{n-1} c_{\pi(i)\pi(i+1)}$ is minimized[4].

Additionally, if the clockwise and anti-clockwise tours are taken as the same tour, the total number of possible tours would yield $\frac{(n-1)!}{2}$, which would be extremely large if n is large. It would be incredibly expensive to compute the total length of all the possible tours and find the best one. Thus various computational algorithms could help to search for the solutions of the TSP.

2.2 2-opt optimization

2-opt is a local search algorithm for solving the TSP. This optimization method is to change the order of the route that crosses over itself into the way it does not. According to this, the permutation could be changed by using a neighborhood operator that reverse all the positions between two randomly chosen indices (m, n), as shown in Figure 2.

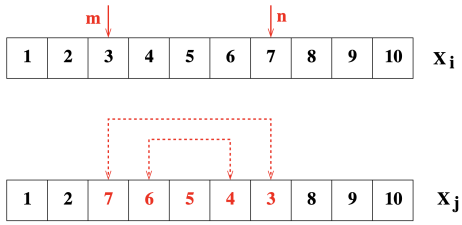

Figure 2: A neighborhood operator: reversing all positions between two randomly chosen positions (m, n)[3].

2.3 Markov chains and the Hasting-Metropolis algorithm

2.3.1 Markov chains. Markov chains are sequences of events that are probabilistically related to each other, where every event depends on the previous event with a given probability. Nevertheless, the Markov chain as a whole is still memoryless.

Let X_i denote the state of the system at time i, the chain after n steps would be $X_0 \rightarrow X_1 \rightarrow \cdots \rightarrow X_n$. Then we could define a matrix of numbers P_{ij}(where $i, j \in [1, 2, \cdots, N]$) in such a way that the probability state i will transit to state j is exactly P_{ij} and get:

$$P_{ij} = P(X_{n+1} = j | X_n = i) \tag{1}$$

$$\sum_{j=1}^{N} P_{ij} = 1 \tag{2}$$

2.3.2 Hasting-Metropolis algorithm . The Hasting-Metropolis algorithm is a common way to construct a time-reversible Markov chain that is easy to simulate. The steps of this algorithm are as followed:

- Choose an irreducible Q with transition probabilities $q(i, j)$. Let $b(j), j = 1, \cdots, m$ be positive values. Also choose k between 1 and m.
- Let $n = 0$ and $X_0 = k$.
- Generate X from $P(X = j) = q(X_n, j)$ and also generate a random number U.
- If $U < \frac{b(X)q(X,X_n)}{b(X_n)q(X_n,X)}$, then $\gamma = X$; else $\gamma = X_n$.
- $n = n + 1; X_n = \gamma$.
- Go to the second step.

According to these steps, we generate a Markov chain starting with sampling the probability space somewhere and probing the next location in the chain by calculating the ratio α, which indicates how probable is the next point compared to the current. Then we accept or reject the next position based on α. If it is more probable than the current point, we move on to this point. If it is less probable, then we generate a random U to decide whether to move or stay. Eventually, this algorithm will move to high probability spaces and converge to the solution.

2.4 Simulated Annealing

Simulated Annealing is a powerful algorithm for global optimization searching and it is useful for solving NP-hard problems like *Traveling salesman problem* . The Simulated Annealing algorithm starts from a random initial state and then operate a random walk in the configuration space. On the one hand, we always accept transitions from the higher energy state to the lower energy state. One the other hand, we would accept transitions from lower the energy state to the higher energy state according to the *Boltzmann probability distribution*. Every step of accepting or rejecting a state is a step of the Hasting-Metropolis algorithm. The detailed algorithm is shown below:

(1) Generate an initial state x_k randomly and initialize the energy function $E(x_k)$
(2) Generate a neighbour state x_{k+1} and calculate the the energy function $E(x_{k+1})$
(3) If $E(x_{k+1}) \leq E(x_k)$ then accept the new state $E(x_{k+1})$
(4) Else accept the transition with probability $\exp -\frac{\delta E}{T}$
(5) Decrease the temperature by a certain cooling schedule
(6) Return back to step (2).

2.5 Cooling Schedule

The cooling schedule is an effective parameter of Simulated Annealing. It determines the procedure on how the temperature decreases. If the temperature drops extremely fast, it is highly likely that we would get stuck in a local minimum; but if it drops too slow, we may not obtain a good result and waste computing power. There are some widely-used cooling schedules available in the literature:[6]

2.5.1 Exponential cooling . Exponential cooling was proposed by Kirkpatrick[5]. The temperature decreases through multiplying the initial temperature T_0 by a factor α concerning the cooling step k:

$$T_k = T_0 \alpha^k (0.8 \leq \alpha \leq 0.99) \tag{3}$$

2.5.2 Linear cooling. For linear cooling, the temperature decrease is made by multiplying the factor α by cooling step k and subtract it from the initial temperature:

$$T_k = T_0 - \alpha k \tag{4}$$

2.5.3 Quadratic cooling. In Quadratic cooling, the initial temperature T_0 is in inverse proportion to the square of cooling step k multiplied by α:

$$T_k = \frac{T_0}{1 + \alpha k^2} \tag{5}$$

3 METHODOLOGY

3.1 Comparison between Cooling schedules

Firstly we would like to compare the effect of different cooling schedules, we implemented three common cooling schedules: exponential cooling, linear cooling and quadratic cooling as mentioned in Equation 3, 4, 5. For the sake of fairness, we have to make sure that the initial temperature, end temperature and the number of cooling steps are identical. Figure 3 visualized the temperature decreasing ratio of three functions. The α of different schedules could be extremely large if it has to meet the same requirement. For example, $\alpha = 0.8$ in exponential cooling could lead to $\alpha = 1.3145 \times 10^{24}$ in quadratic cooling, which is unacceptable. Thus we chose the three α, given in Figure 3, in our experiment.

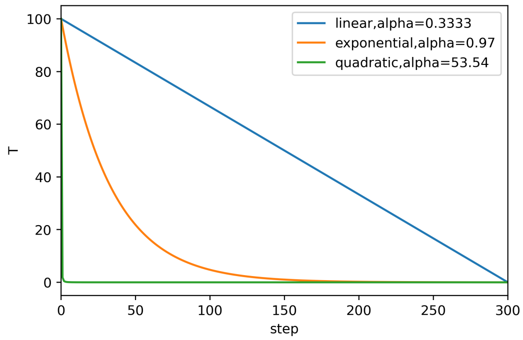

Figure 3: Three cooling schedules with same initial temperature and end temperature. The number of cooling steps are 300.

3.2 Different initial temperature, cooling ratios

Then we investigated the effect of initial temperatures, number of cooling steps and different cooling ratio α. We chose the exponential cooling schedule from now on because its cooling ratio is moderate and relatively easy to control, compared with the other two schedules. In the first experiment, we set the $\alpha = [0.80, 0.85, 0.90, 0.95, 0.99]$ to see how this parameter would affect the process of finding minimum. Afterwards, we tried different initial temperatures $T_0 = [10, 50, 100, 150, 200]$. Besides, we always kept our mind on the effect of cooling steps. During a 2000-cooling-steps test, five sets were picked out, namely 150, 300, 600, 1000, 2000.

3.3 Effects of Markov chain's length

The Markov chain's length would determine how many times the state would change before the temperature went down. We would like to evaluate the effects of the Markov chain's length on the convergence of solutions. Thus we set different values of Markov chain's length as [50,100,150,200,250], keep other parameters the same and operate the SA algorithm with the exponential cooling schedule.

3.4 Final solutions for two sets

After operating the experiments mentioned above, we could finally determine the parameters we could use to search for the optimal solutions of the set of 280 cities. Intuitively, we would have to balance our computing power and the probabilities of finding a better solution with larger parameters. Additionally, we tried to address a bigger problem with the set of 442 cities and searched for the best solution we could obtain.

4 RESULTS AND DISCUSSION

4.1 Effects of cooling schedules on convergence

The convergence speed towards local/global minimum depended on the cooling schedule. Figure 4 compared the optimization searching process of three given cooling schedules. It was clear that quadratic cooling had the best result, followed by exponential cooling and linear cooling. The local minimum found by these schedules is 7821.52 ± 257.89, 4643.18 ± 133.79, 3548.66 ± 127.44 respectively.

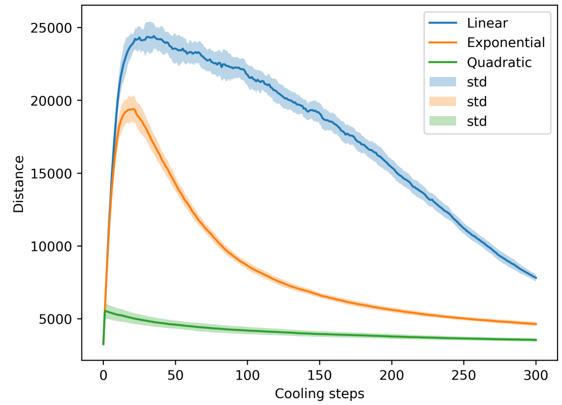

Figure 4: Comparison between different cooling schedules. The cooling ratio of linear, exponential, quadratic is 0.333, 0.97, 53.54 respectively. $T_0 = 100$, cooling steps=300, Markov chain's length=100, simulation=30.

Besides local minimum, we noticed that exponential cooling converged fastest in the first half, meanwhile linear and quadratic converged relatively slower. As we mentioned in Figure 3, quadratic cooling drops most rapidly while linear cooling drops slowest. Having a longer high-temperature period, linear cooling had more opportunity jumping to neighbour state with higher energy level, but it seemed hard to convergence with the limited 300 cooling steps. The fluctuation between 40-150 steps also indicated such process. In contrast to linear cooling, quadratic cooling dropped too fast, causing it stuck in the local minimum. Apart from the convergence rate, we also found that quadratic cooling was hard to control if given the same initial temperature and end temperature. If we set $\alpha = 0.8$ for exponential cooling, we would consequently get $\alpha = 1.3145\times10^{24}$ for quadratic cooling, which is unacceptable in practice. Thus far, we investigated the performance of three cooling schedules and we decided to use exponential cooling in the later experiment.

4.2 Effects of initial temperature, cooling steps and ratios

The second test was performed using 5 different initial temperatures. Figure 5 showed the minimum searching process and Figure 6 showed the final result. As we can see, the processes with relatively higher initial temperature 100, 150, 200 had similar convergence behaviour: the cost function (total distance) boosted in the beginning and dropped rapidly with a decreasing convergence rate. The other two, by contrast, did not have significant change during the whole process. That is to say, we stuck in the local minimum and rarely had chances to accept the state with a larger distance. Although the best result 2997.92 ± 70.23 was obtained by the lowest initial temperature 10, we did not use this temperature afterwards because we reckon more cooling steps would lead to a better result for higher initial temperatures.

We found that the number of cooling steps is the key factor affecting the final result. It was clear in Figure 7 and Figure 8 that

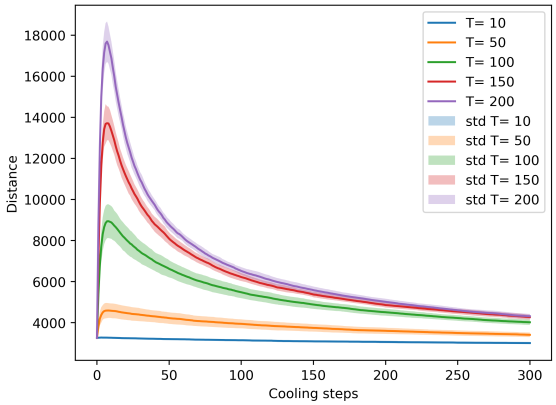

Figure 5: The effect of different initial temperatures. Exponential cooling $\alpha = 0.8$, cooling steps=300, Markov chain's length=100, simulation=30.

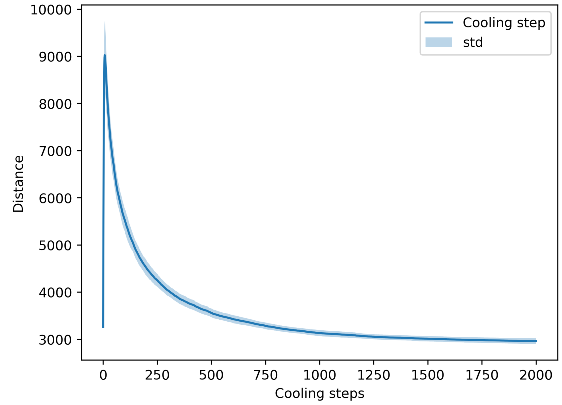

Figure 7: The effect of number of cooling steps. Exponential cooling $\alpha = 0.8$, maximum cooling steps=2000, $T_0 = 100$, Markov chain's length=100, simulation=30.

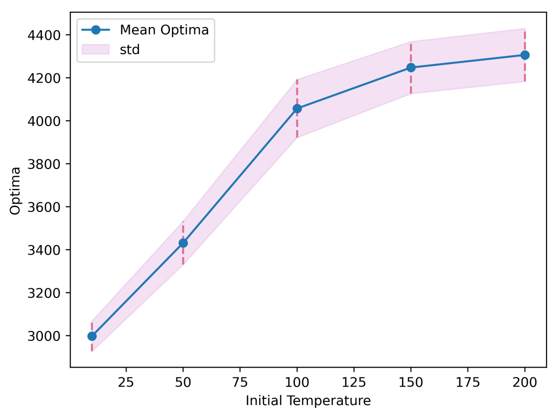

Figure 6: The local minimum of different initial temperatures. Exponential cooling $\alpha = 0.8$, cooling steps=300, Markov chain's length=100, simulation=30.

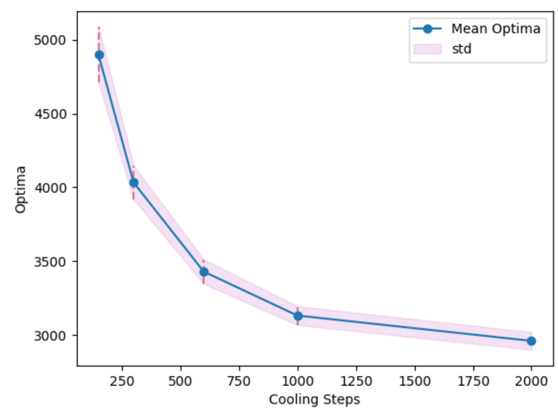

Figure 8: The local minimum at five cooling steps. Exponential cooling $\alpha = 0.8$, maximum cooling steps=2000, $T_0 = 100$, Markov chain's length=100, simulation=30.

the increasing cooling step would cause the distance, at least within our computing limitation, decreasing continuously. In Figure 8, the total distance still tended to decrease further even if the cooling steps was 2000; but we did notice that the decreasing rate and standard deviation shrank. We picked five standard deviations at five cooling steps(150, 300, 600, 1000, 2000) and they were 187.73, 114.06, 82.06, 62.52, 59.51, which has a decreasing trend in general.

With regard to cooling ratio, Figure 9 and Figure 10 revealed the differences between different α. Except for the highest $\alpha = 0.99$, the optimum and standard deviation of the remaining tests were very close. Although results were close, we still found that the smallest $\alpha = 0.8$ had the best performance.

4.3 Effects of Markov Chain's length on convergence

As shown in Figure 11, the convergence occurred in every set of Markov chain's length. The mean distance increased significantly when the temperature was still high in the early stage of cooling, then dropped as the temperature went down and became relatively steady at last. Additionally, the standard deviation also converged to a low value at the end of the annealing process for every set of Markov chain's length. Interestingly, we noticed that the larger Markov chain's length was, the peak of distance was higher but the final distance was lower, which was due to more sampling at high temperature and more acceptance of larger values. The more search space we sampled, the higher probabilities we could find the global minimum. Figure 12 also revealed that the final optima decreased

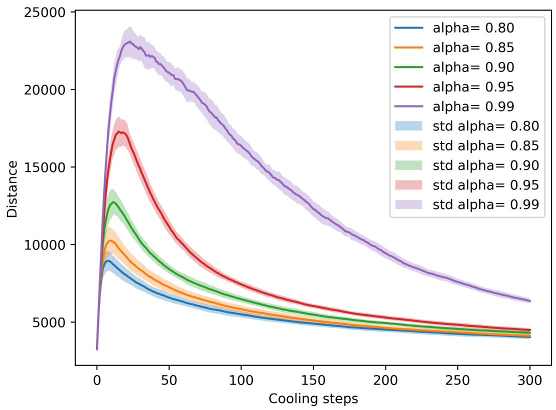

Figure 9: The effect of different cooling ratio α of exponential cooling, ranging from 0.8 to 0.99. Cooling steps=300, $T_0 = 100$, Markov chain's length=100, simulation=30.

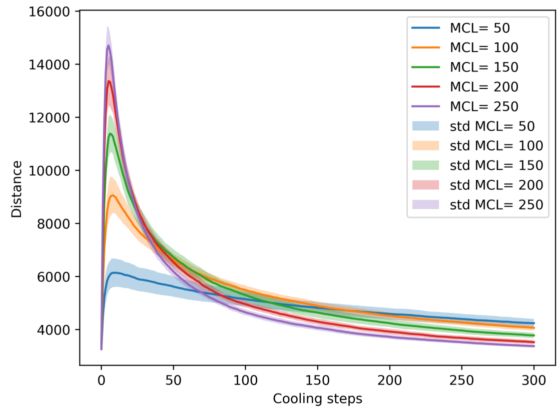

Figure 11: The effect of Markov chain's length on convergence, $T_0 = 100$, cooling steps=300, exponential cooling $\alpha = 0.8$, simulation=30.

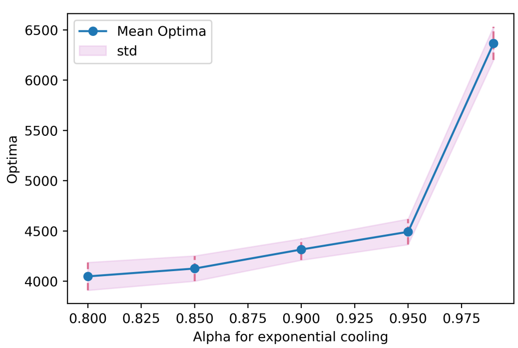

Figure 10: The local minimum found by exponential cooling using different α. Cooling steps=300, $T_0 = 100$, Markov Chain length=100, simulation=30.

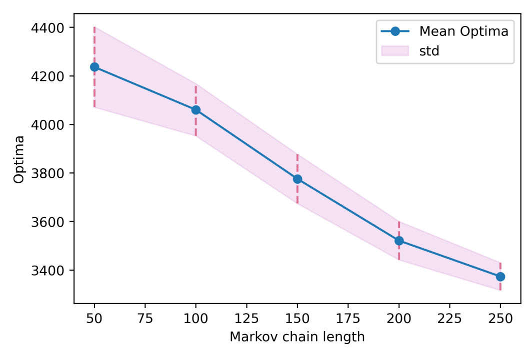

Figure 12: The local minimum of different Markov chain's length, $T_0 = 100$, cooling steps=300, exponential cooling $\alpha = 0.8$, simulation=30.

as the Markov chain's length grew, with standard deviation going down.

Intuitively, the optima would converge to a stable value if the Markov chain's length is sufficiently large. We could not reach this point because of limited computing power. However, we could still conclude that increasing the Markov chain's length would result in better solutions.

4.4 Local minimum for two sets

After all the tuning process of different parameters, we could finally determine the setting for finding a new minimum. Although some parameters should be sufficiently large to obtain better solutions, we could not reach them due to limited computing power. Thus we could only converge to a local minimum rather than the global minimum.

4.4.1 Set of 280 cities. Given the set of 280 cities, we set the initial temperature as 100, cooling steps as 2000, Markov chain's length as 100 and used exponential cooling with $\alpha = 0.8$ to search for the local minimum. With 30 simulations, we obtained the minimal distance was 2962.48 ± 59.51, whose convergence was shown above in Figure 7. This solution has a certain distance with the global minimum of 2586.77 but still the best one we could get. Figure 13 showed the final route of 280 cities in one of the simulations.

4.4.2 Set of 442 cities. Likewise, given the set of 442 cities, we set the initial temperature at 100, cooling steps at 2000, Markov chain's length at 250 and used exponential cooling with $\alpha = 0.8$ to search for the local minimum. Due to the limited computing power, we only ran the simulation once and got the minimal distance of 57969.0144 and this final route was shown in Figure 14.

5 CONCLUSION

In summary, we implemented Simulated Annealing to figure out the Traveling salesman problem. The cooling schedule is a key factor in this algorithm, thus we first compared the convergence behaviours of linear cooling, exponential cooling and quadratic cooling and found that exponential cooling is more suitable in our experiment.

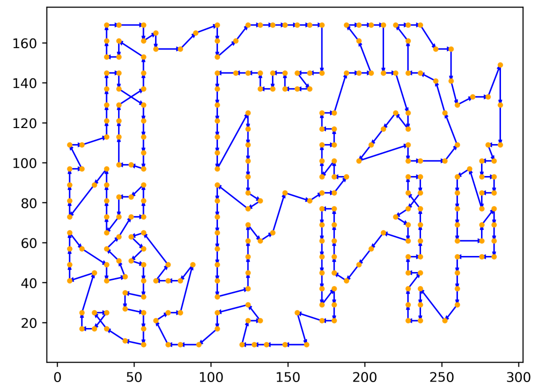

Figure 13: Final route for 280 cities, $T_0 = 100$, cooling steps=2000, exponential cooling $\alpha = 0.8$, Markov chain's length=100.

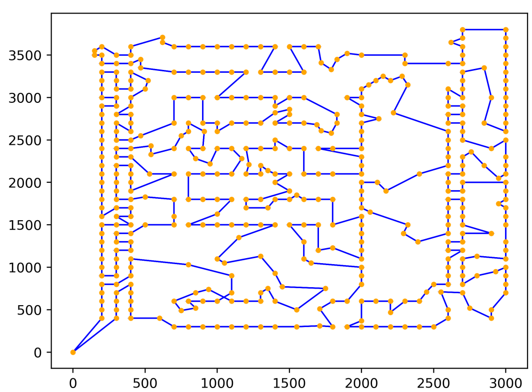

Figure 14: Final route for 442 cities, $T_0 = 100$, cooling steps=2000, exponential cooling $\alpha = 0.8$, Markov chain's length=250.

Then we investigated the effect of initial temperature, the number of cooling steps, cooling ratio α and Markov Chain length. The results showed that a higher initial temperature did not necessarily improve our final result; but if the initial temperature is low, we are not likely to obtain a good result. However, increasing the cooling steps, Markov chain's length and having a faster cooling ratio would improve the optima to a certain degree. Finally, we found the shortest distance for 280 cities was 2962.48 ± 59.51 and for 442 cities was 57969.0144. In future study, we could try other state-of-the-art cooling schedules and use a more powerful computer to operate more simulations and eventually reach the equilibrium state with sufficient cooling steps and Markov chain's length.

REFERENCES

[1] Dimitris Bertsimas, John Tsitsiklis, et al. 1993. Simulated annealing. *Statistical science* 8, 1 (1993), 10–15.

[2] Wikipedia contributors. 2019. Traveling salesman problem — Wikipedia , the free encyclopedia. https://en.wikipedia.org/w/index.php?title=TravelTravelerProblem&oldid=54534603 [Online; accessed September 9, 2019].

[3] Daniel Delahaye, Supatcha Chaimatanan, and Marcel Mongeau. 2019. Simulated annealing: From basics to applications. In *Handbook of Metaheuristics*. Springer, 1–35.

[4] Gregory Gutin and Abraham P Punnen. 2006. *The traveling salesman problem and its variations*. Vol. 12. Springer Science & Business Media.

[5] Scott Kirkpatrick, C Daniel Gelatt, and Mario P Vecchi. 1983. Optimization by simulated annealing. *science* 220, 4598 (1983), 671–680.

[6] José Fernando Díaz Martín and Jesús M Riaño Sierra. 2009. A comparison of cooling schedules for simulated annealing. In *Encyclopedia of Artificial Intelligence*. IGI Global, 344–352.