Fundamentals of Data Structures and Algorithms for AI

Simulated Annealing and Parallel Tempering on Travel Salesman Problem

Prepared By

Sintayew Zekarias and Fikir Awoke

December 19, 2021

Course Instructor - Dr. Beakal Gizachew

Addis ababa Institute of Technology

School of Information Technology and Engineering

Table of Contents

1. Introduction

A number of algorithms are used to obtain an optimal solution for different problems having a huge search space. The algorithms can be categorized as Informed (heuristic) and Uninformed (blind) algorithms based on their search strategies. Simulated annealing (SA) is one of the most popular and intelligent optimization algorithms which is categorized in heuristic algorithms. These algorithms use a problem-specific evaluation and neighborhood function. Simulated annealing algorithm, which was first independently presented as a search algorithm for combinatorial optimization problems, is a popular iterative metaheuristic algorithm widely used to address discrete and continuous optimization problems. The other effective algorithm is the parallel tempering which is also used for combinatorial optimization. Parallel tempering (PT), also called replica exchange or simulated tempering is a Monte Carlo method intended primarily for sampling a probability distribution function with a complex structure. This paper proposes an effective local search algorithm based on simulated annealing and parallel tempering techniques to solve the travelling salesman problem (TSP). The travelling salesman problem is when given a list of cities and the distances between each pair of cities, the shortest possible route is to visit each city exactly once and return to the origin city. It is an NP-hard problem in combinatorial optimization. Our objective in this paper is to implement parallel tempering and simulated annealing on the traveling salesman problem. We will compare the performance of the two and also show how different choices of parameters affect the simulation results and performance.

2. Methodology

2.1. Simulated Annealing

The Concept of simulated annealing came up from the physical annealing process of heating a metal to a high temperature and holding it there for a certain length of time, and then letting it cool down slowly to achieve minimal internal energy. Based on this concept, simulated annealing uses a schedule of temperatures to solve optimization problems.

In the simulated annealing algorithm, new points are always accepted if they are better than the current point, and when the new points are worse there will be a probability of acceptance. The following figure illustrates the simulated annealing algorithm.

```
procedure simulated annealing
begin
   t ← 0
   initialize T
   select a current point v_c at random
   evaluate v_c
   repeat
      repeat
         select a new point v_n
            in the neighborhood of v_c
         if eval(v_c) < eval(v_n)
            then v_c ← v_n
            else if random[0,1) < e^((eval(v_n) − eval(v_c))/T)
                 then v_c ← v_n
      until (termination-condition)
      T ← g(T, t)
      t ← t + 1
   until (halting-criterion)
end
```

For implementing the travel salesman problem, we generated 150 random coordinates that correspond to the cities. We initialized the necessary variables and defined a variable that holds the index of all coordinate values starting from 0 to the length of the coordinates. We used the Greedy algorithm to get an initial solution (closest-neighbour). These are the steps used to implement simulated annealing on TSP:

1. Selection of the current node at random, which will be the starting point.
2. Find the nearest neighbour for the current node, by calculating the euclidean distance between the current node and every other node.
3. Comparison is made between the current fitness and best fitness, if the best fitness is greater it will be taken as the current fitness.
4. Calculation of the total distance is made from the weight of the path.
5. Then the probability of acceptance is made. If the candidate is worse than the current, probability of acceptance depends on the current temperature and the difference between candidate and current. And If the candidate is better than the current one, it will be accepted with probability 1.
6. Visualizing the travel salesman problem is made based on the path.
7. The fitness is plotted through the iterations made.

2.2. Parallel Tempering

Applying multiple temperatures in a simulation is crucial for sampling the configuration space with a complex structure. In a simulated annealing, a fixed schedule of temperatures is used, and in a parallel tempering rather than jumping between temperatures, it simultaneously simulates multiple chains, each at a temperature level Tk, called a replica, and constantly swaps samples between replicas. Each iteration consists of the following steps:

1. Parallel update : simulates each replica with its own transition kernel
2. Replica exchange : it is a swap state between two replicas which will be based on the probability of acceptance.

The following figure illustrates the parallel tempering algorithm.

for $i \leftarrow 0$ **to** $M-1$ **do**
  $T_i \leftarrow$ calculate_temperature(i)
  $replica_i \leftarrow$ new instance of the ACST with T_i
end
while *Not reached stopping criterion* **do**
  for $i \leftarrow 0$ **to** $M-1$ **do in parallel**
    Run $replica_i$ for N_{swap} iterations
  end
  $G \leftarrow min(Cost(global\_best_{replica(0)}), \ldots, Cost(global\_best_{replica(M-1)}))$
  $j \leftarrow \mathcal{U}\{0, M-2\}$ `// Randomly select two adjacent temp. levels`
  $\Delta E \leftarrow (Cost(active\_solution_{replica(j)}) - Cost(active\_solution_{replica(j+1)}))/G$
  $\Delta\beta \leftarrow \frac{1}{T_j} - \frac{1}{T_{j+1}}$
  if $\mathcal{U}(0,1) < min(1, \exp(\Delta\beta\Delta E))$ **then**
    Set the temp. of $replica_j$ and $replica_{j+1}$ to T_{j+1} and T_j, respectively
    swap($replica_j, replica_{j+1}$) `// Keeps replicas ordered by temp.`
  end
end

3. Simulation Results

For our simulations, we used an instance of 150 cities. The following figure shows the city coordinate distribution.

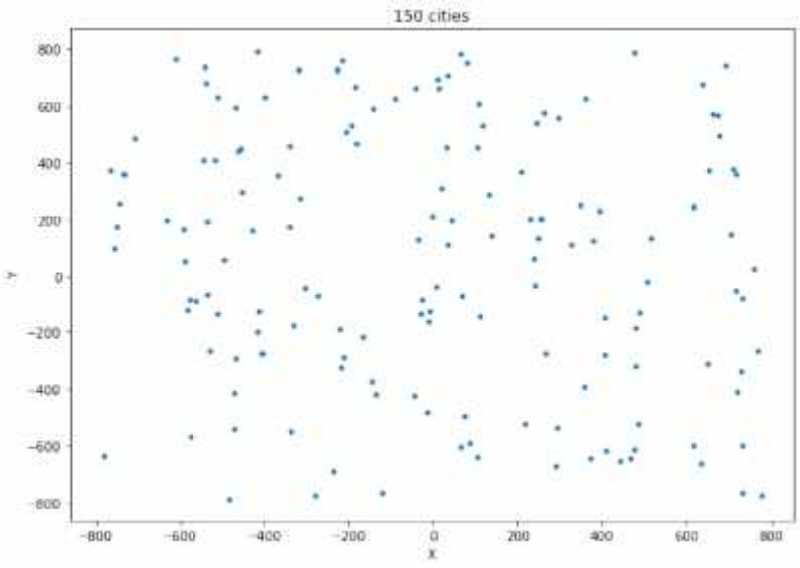

Figure 1: City Coordinate distribution

3.1. Before Simulated Annealing

Before the annealing process for the travel salesman problem, with the parameter of stopping Iteration = 20000 the initial best fitness obtained is = 17329.658272641176.

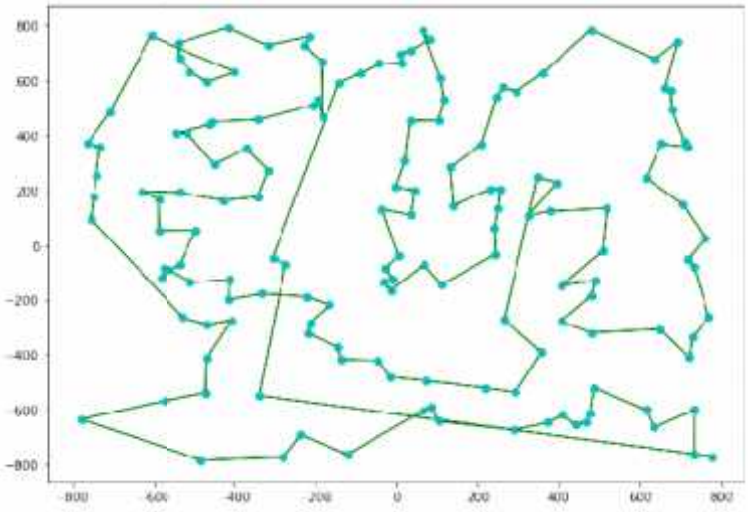

Figure 2: Before simulated annealing with 20000 iteration

With parameter of stopping Iteration = 80000
Initial best fitness obtained: 19369.318462383904

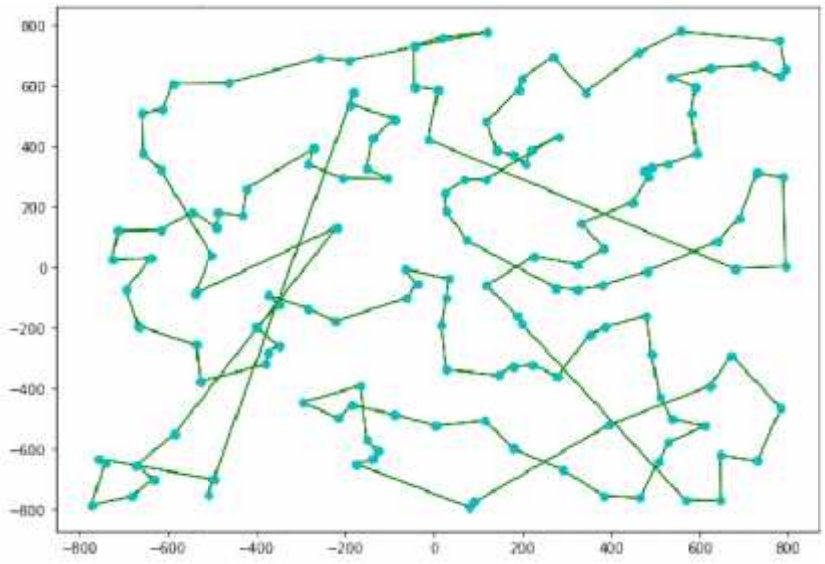

Figure 3: Before simulated annealing with 80000 iteration

3.2. Simulated Annealing

Using the implementation of Section 2.1, we run simulated annealing on our n = 150 instance with parameter of stopping Iteration = 20000, T = square root of number of nodes, alpha = 0.995 and stopping_temperature = 1e-8 and got the following result for the travel salesman problem:

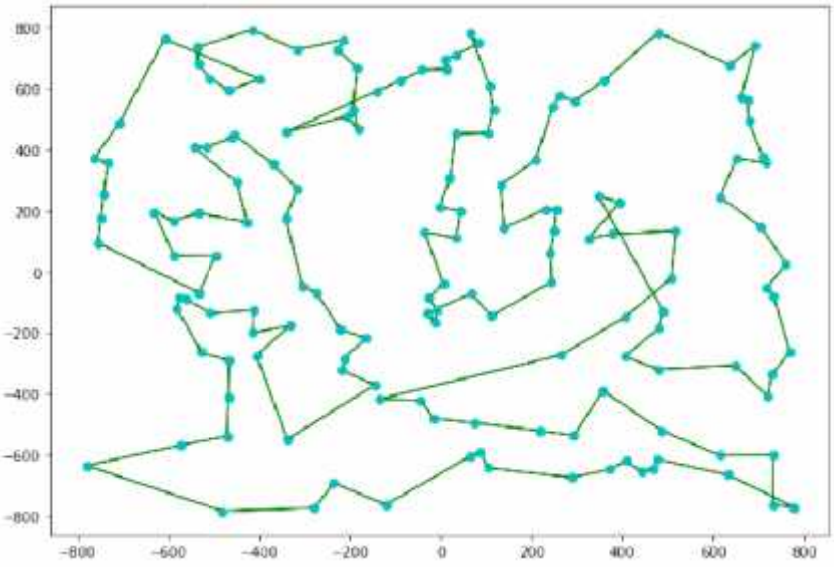

Figure 4: Simulated Annealing Result with 20000 iteration

After simulated annealing:
Final best fitness obtained: 16536.178207123696
Improvement over greedy algorithm: 4.58%

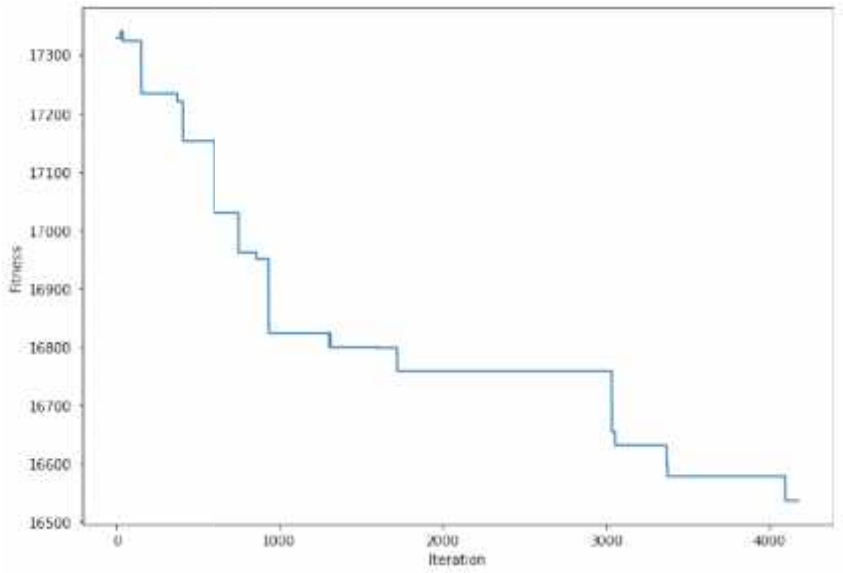

Figure 5: Simulated Annealing with 8000 iteration

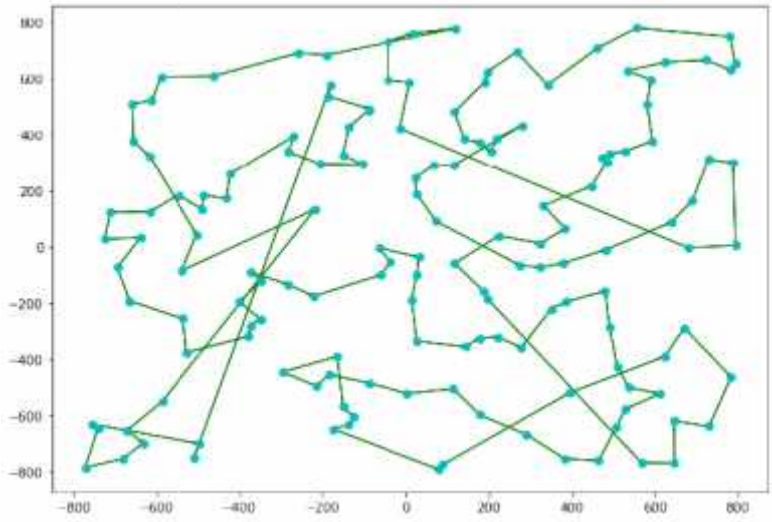

Figure 6: Simulated Annealing Result with 20000 iteration

With a parameter of stopping Iteration = 80000,
Final best fitness obtained: 17865.198756030302
Improvement over greedy heuristic: 7.77%

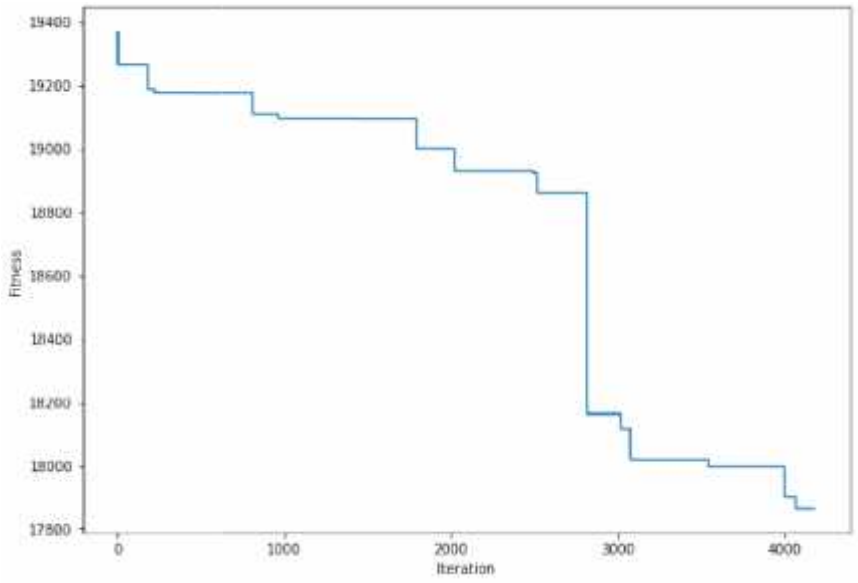

Figure 7: Simulated Annealing with 80000 iteration

3.3. Parallel Tempering

Using the implementation of Section 2.2, we run parallel tempering on our n = 150 instance with parameter of stopping Iteration = 60000, Nswap = 1, number of processes = 4 , Initial Temperature = the number of processes, cooling ratio = 1/ratio, ratio = the square root of the number of processes, number of run = 15 and got the following result for the travel salesman problem:

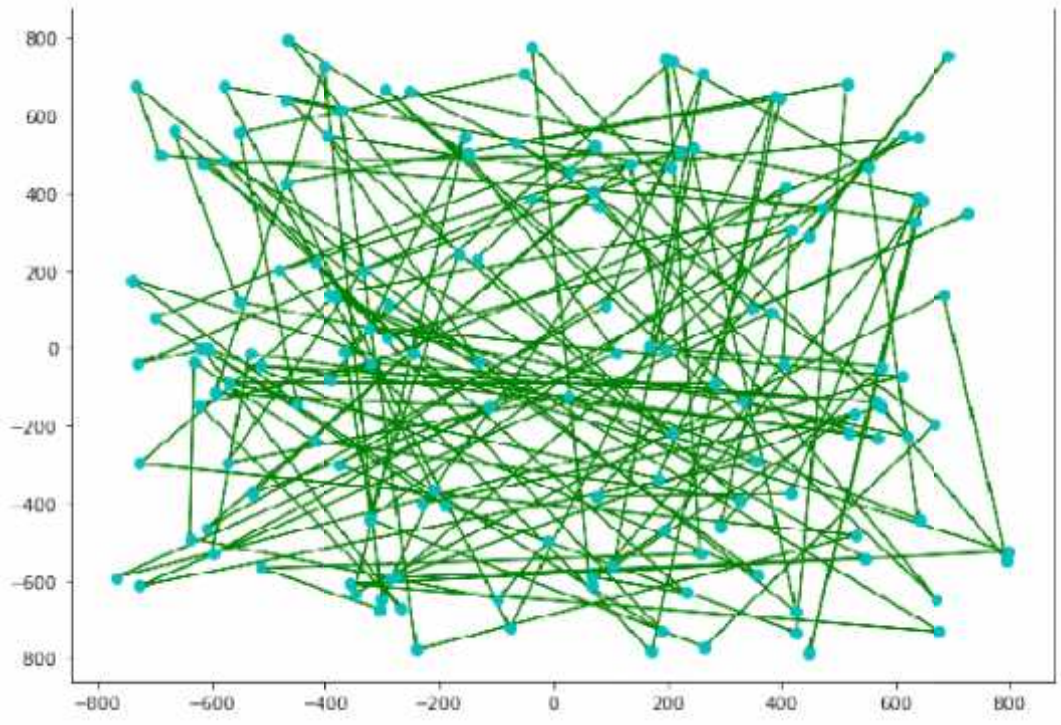

Figure 8: Parallel Tempering Result

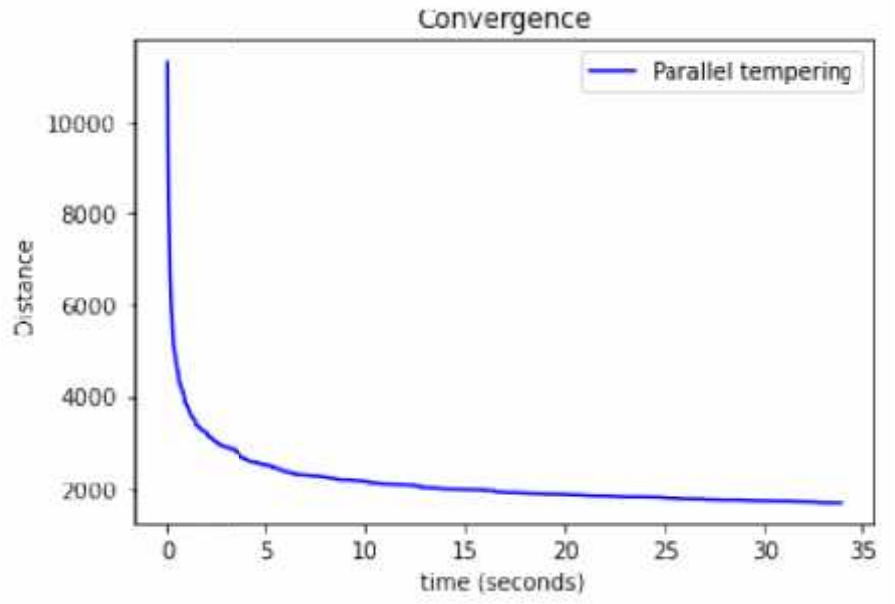

Figure 9: Parallel Tempering convergence

5. Conclusion

We have implemented the Simulated Annealing and Parallel Tempering algorithms on the Travel Salesman Problem by generating random coordinates for the cities. For the simulated annealing algorithm we used different parameters to check the performance. Unlikely in the case of parallel tempering, we had a hard time understanding the concept behind it. Despite all the confusion, we tried to implement it based on our level of understanding. While searching and reading for this project we got a lot of knowledge on previous and current optimization algorithms.

References

https://www.sciencedirect.com/science/article/abs/pii/S1568494611000573
https://www.hindawi.com/journals/cin/2016/1712630/
https://www.researchgate.net/publication/243358870_Parallel_Tempering_for_the_Traveling_Salesman_Problem
https://www.slideshare.net/lindahua2015/mlpi-lecture-3-advanced-sampling-techniques