Lab Tutorial For Solving TSP using Simulated Annealing

Kaishav Basodiya, *202051094*
Amanshu jaiswal, *202051023*
Aditya Priyanshu, *202051010*
and Abhishek Kumar, *202051003*

Abstract—Simulated Annealing is a popular optimization technique that has been successfully applied to various optimization problems, including the famous Travelling Salesman Problem (TSP). The TSP is an NP-hard problem that involves finding the shortest possible route that visits a set of cities and returns to the starting city. In this paper, we try to get optimal path of TSP using the Simulated Annealing algorithm. The algorithm starts with an initial random tour and iteratively updates the tour by making small random changes until a near-optimal solution is found. The acceptance of new tours is based on a probability function that balances the exploration and exploitation of the search space. The proposed method is tested on various benchmark TSP datasets and results demonstrate the effectiveness and efficiency of the Simulated Annealing approach.

Index Terms—TSP ,NP-hard, Simulated Annealing, LaTeX.

1 Introduction

The Travelling Salesman Problem (TSP) is a classic optimization problem in computer science. In this we try to find the shortest possible route that visits a set of cities and returns to the starting city. The challenge of the TSP lies in finding the optimal solution among a large number of possible solutions, as the number of possible tours grows rapidly with the number of cities. As a result, the TSP is considered an NP-hard problem, meaning that finding an exact solution for large instances of the problem can be computationally expensive.The TSP has many real-world applications, such as scheduling, logistics, and vehicle routing.

1.1 Different Approaches For Optimal Solution Of TSP-

There are several approaches to solve the Travelling Salesman Problem (TSP), including exact algorithms and heuristic algorithms. Exact Algorithms guarantees to give optimal solution but they are computationally expensive example - Branch & Bound , Dynamic Programming.
Heuristic Algorithms are faster but optimal solution isn't guaranteed Example - Simulated Annealing, Ant-colony Optimization
There are other approaches as well like christofides Algo, 2/3-opt Local search,etc.

Simulated Annealing -

Simulated Annealing (SA) is a heuristic optimization algorithm that can be used to solve the Travelling Salesman Problem (TSP). It is a probabilistic method that is inspired by the annealing process used in metallurgy to refine metals.

In the TSP, the algorithm starts with an initial random tour and iteratively updates the tour by making small random changes. The acceptance of new tours is based on a probability function that balances the exploration and exploitation of the search space. The algorithm gradually reduces the temperature, which determines the probability of accepting a new solution that is worse than the current one. This allows the algorithm to escape from local optima and explore the search space more effectively. The algorithm terminates when the temperature has cooled to a certain level or when a satisfactory solution is found.

SA has been shown to perform well on many TSP instances and has been used in various applications, such as vehicle routing and scheduling. Its advantage lies in its ability to balance the exploration and exploitation of the search space and find near-optimal solutions in a reasonable amount of time. However, SA can be computationally expensive and may not converge to the optimal solution in all cases. The choice of parameters, such as the initial temperature and cooling schedule, can also impact the performance of the algorithm.

Pseudo Code For Simulated Annealing -

We can define some functions which would help in understanding the code of TSP using Simulated Annealing.
generate_random_tour() - takes input of set of cities and generate a random tour
swap_tour() - swaps the order of 2 random cities in current tour
tour_cost() - calculate the total cost of input tour

Now We can define the pseudocode as follows-

```
SimulatedAnnealing(cities,initial_temperature,
cooling_rate,t_min){

    current_tour = generate_random_tour(cities)
    best_tour = current_tour
    temperature = initial_temperature

    while temperature ≥ t_min
        new_tour = swap_tour(current_tour)
          delta_E = evaluate_tour(new_tour) -
        evaluate_tour(current_tour)
        if delta_E ≤0{
            current_tour = new_tour
              if evaluate_tour(current_tour) ≤
            evaluate_tour(best_tour)
            best_tour = current_tour}
        else{
            prob = e^(-delta_E/temperature)
            if random(0, 1) ≤ prob{
                current_tour = new_tour}}
        temperature *= cooling_rate

    return best_tour
}
```

Actual Code and Results-

For Code and Results ,You can visit our Code repository

Conclusion

As Simulated Annealing is a heuristic based Approach so to get optimal solution is not sure and our results are also supporting the fact. We tried to get better solution by applying greedy algorithm to find a tour on cities and then we applied Simulated Annealing on that tour and we get better solution with lesser time. We also observed that if we increase the iterations and tour was becoming more optimized so Simulated Annealing can give optimal solution if we increase its iteration but we were not sure for how much iteration the solution would be the optimal one.

Acknowledgments

The authors would like to thank Prof. Pratik Shah *IIIT Vadodara ,*

References

[1] A First Inntroduction to Artificial Intelligence By Prof.D Khemani IIT Madras.