DONE UNDER

Dr.NABAJYOTI MAZUMDAR SIR

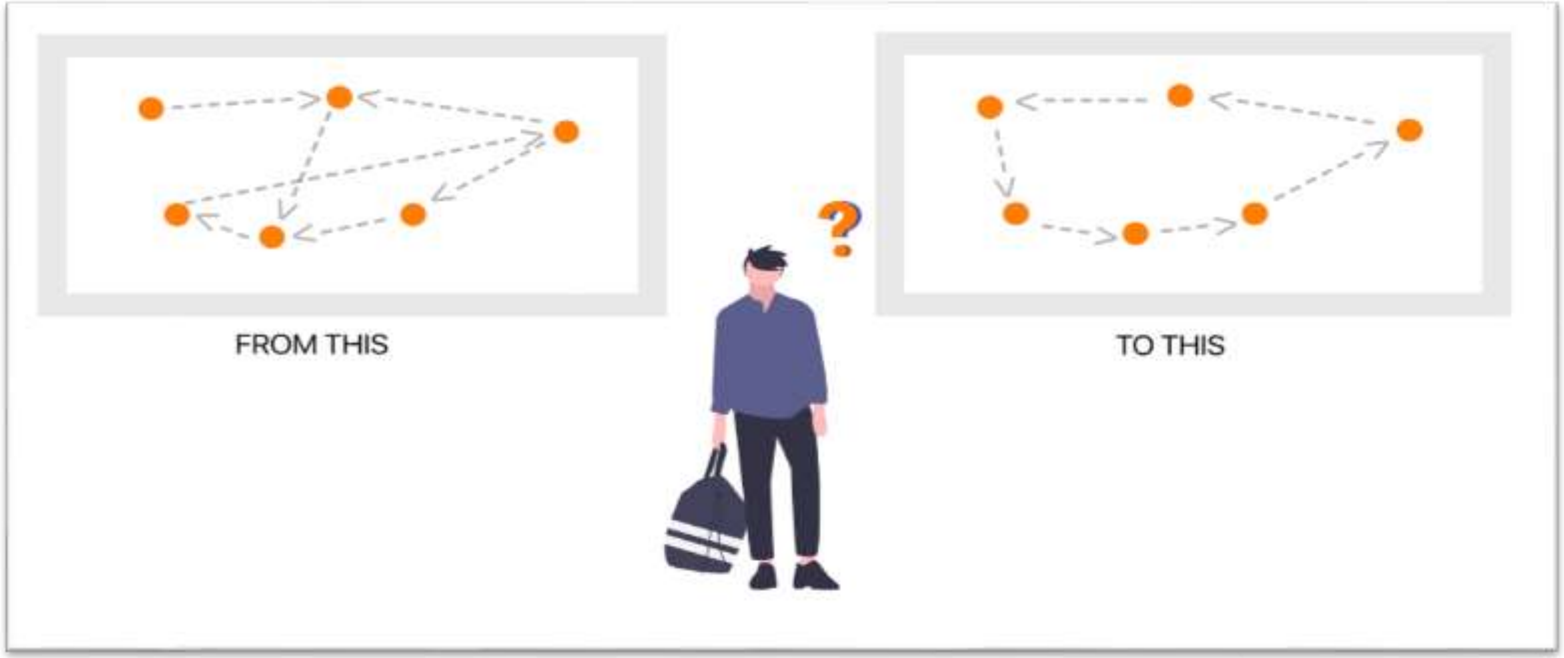
PROJECT MEMBERS

Jayendra Reddy(IIT2021268)

Vignesh(IIT2021262)

Harsha Vardhan Raju(IIT2021258)

EXPLORING VARIOUS HEURISTICS FOR SOLVING CLASSIC AND VARIANT OF TSP



A quick glance on the work till the previous evaluation

- Studied about the classical TSP.
- Found the lower bound using a dataset.

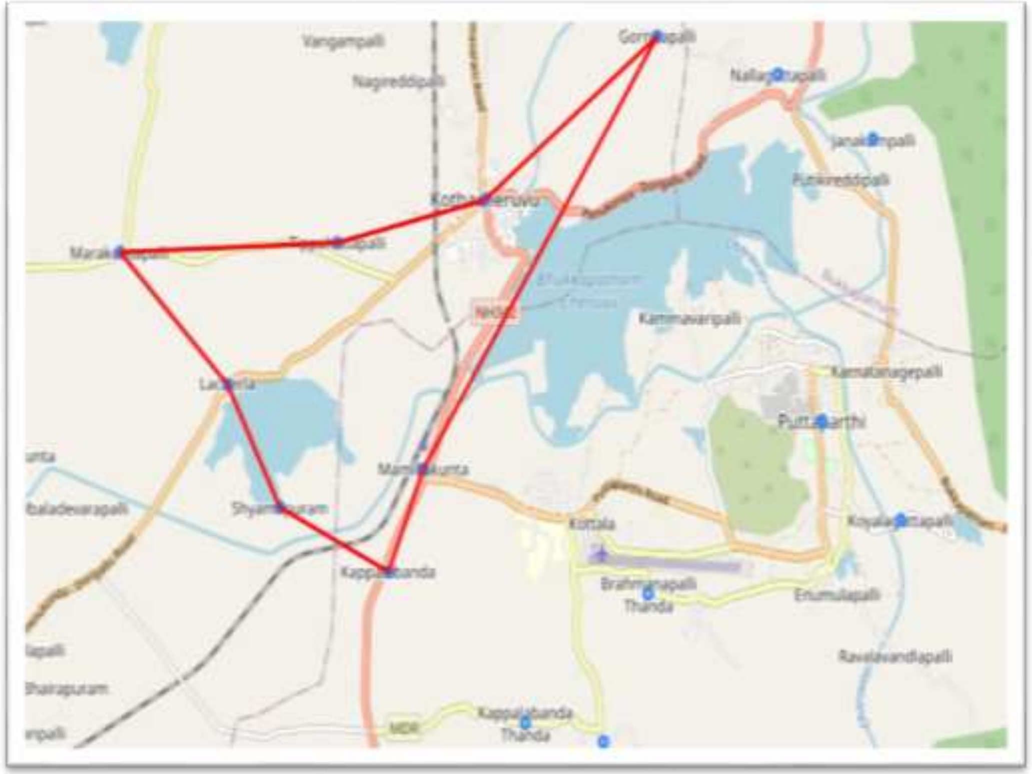
```
mamillakunta
lacherla
puttaparthi
shyamapuram
gorntlapalli
nallaguttapalli
janakampalli
kothacheruvu
kappalabanda
marakuntapalli
tippahatlapalli
koyalaguttapalli
brahmanapalli thanda
kappalabanda thanda
bidupalli
```

```
[0, 8, 3, 1, 9, 10, 7, 4, 5, 6, 2, 11, 14, 13, 12, 0]
TSP cost : 35.92
MST cost : 28.45
Max-one-tree-cost : 32.25
```

- Employed two heuristics to solve the Classical TSP :
  1)Genetic Algorithm
  2)Ant Colony Optimization(ACO)
- Compared the results obtained and found that the ACO gives better results.

- Implemented a variant of the TSP and applied ACO on it.

Quota TSP:




```
quotas : [83, 61, 79, 65, 61, 85, 77, 91, 55, 81, 55, 78, 95, 90, 67]
target quota : 500
path_generated : [0, 8, 3, 1, 9, 10, 7, 4, 0]
path distance : 21.49
```

Overview of PPT:

- Implemented two more heuristics –
  1)Simulated Annealing and
  2)Christofides Algorithm
- Hyperparameter tuning of the ACO parameters using –
  1)Bayesian Optimization
  2)Response Surface Methodology
- Limitations of the algorithms we explored.
- Brief introduction about Reinforcement Learning
- Implementation of Reinforcement Learning on classical TSP.
- Reinforcement Learning on a variant of the TSP(Quota)
- Implemented a practical scenario using :
  1)Q-Learning
  2)SARSA

Simulated Annealing:

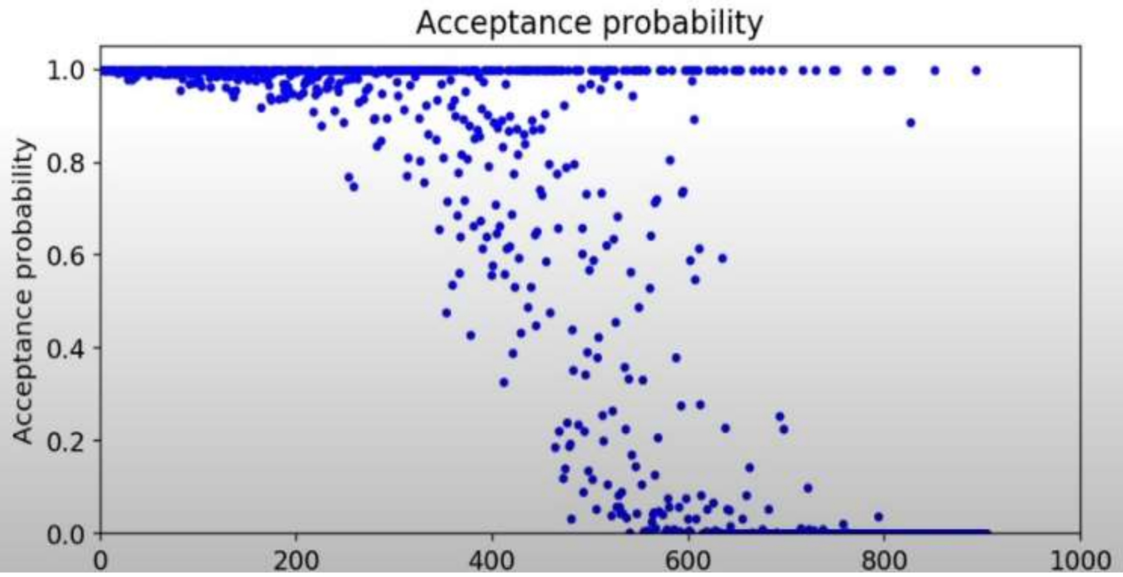
- An optimization algorithm inspired by the annealing process in metallurgy.
- Starts with an initial combination of cities which can be a random one.
- Iteratively explores the solution space by making small changes to the current solution.
- Accepting the new solution would be based on the Metropolis criterion.
- Parameters :

a) Temperature -

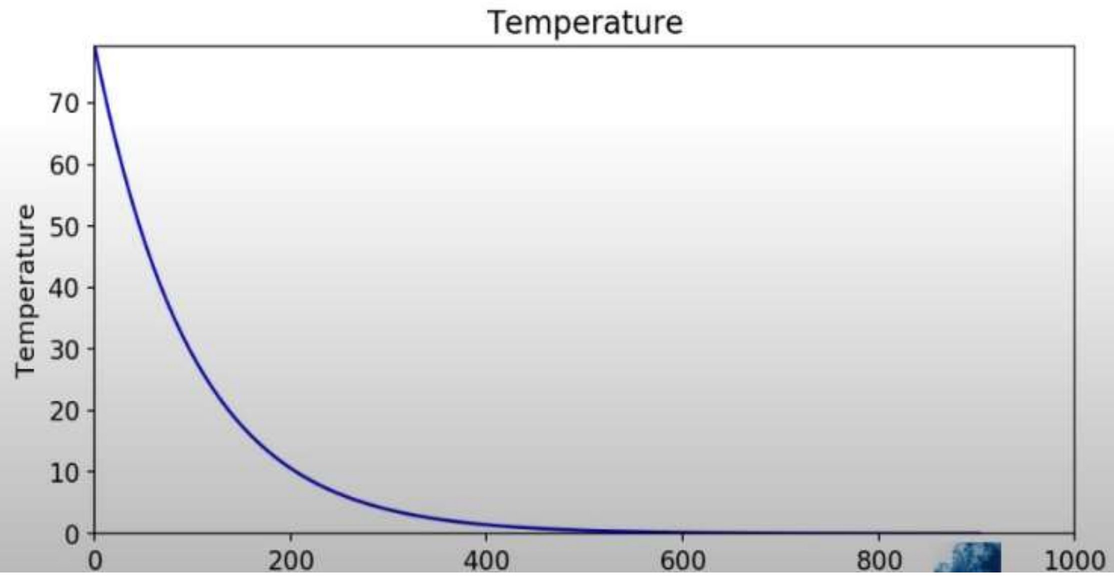
- It determines the probability of accepting worse solutions early in the process .
- Initially, when the temperature is high, the algorithm has a higher probability of accepting worse solutions, allowing it to escape local optima.
- As the temperature decreases, the probability of accepting worse solutions decreases, leading the algorithm towards convergence to an optimal or near-optimal solution.

b) Cooling Rate –

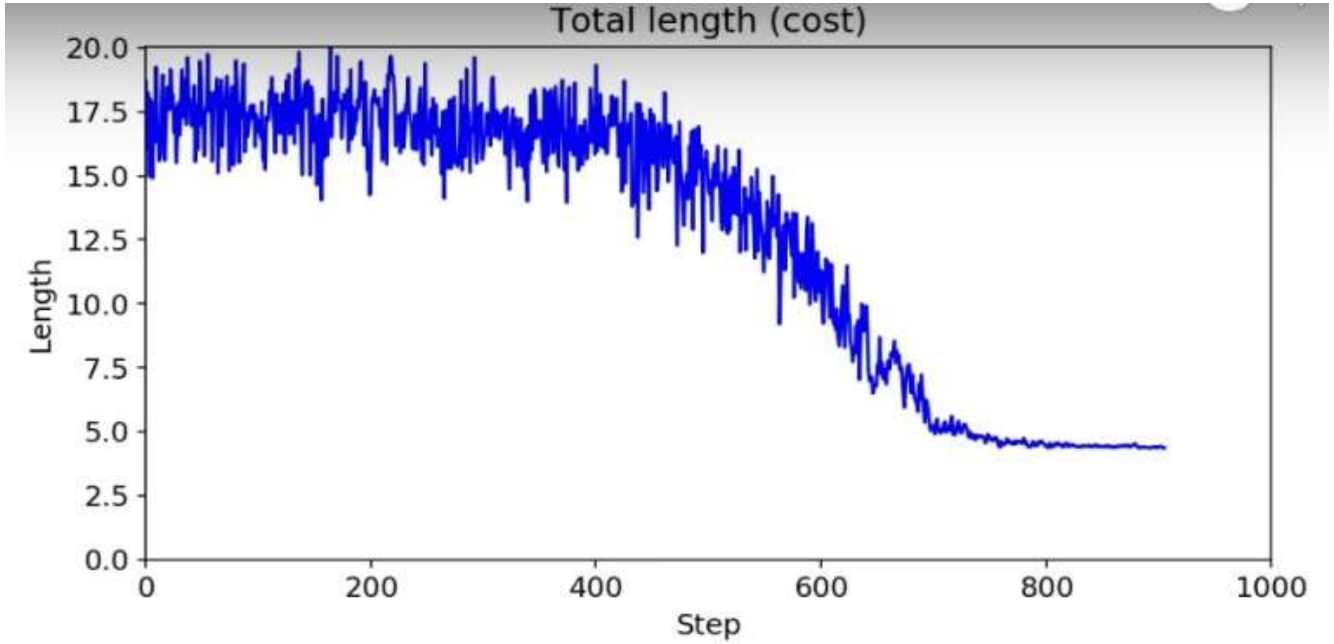
- It controls the rate at which the temperature decreases during the annealing process.



Acceptance probability
Acceptance probability
1.0
0.8
0.6
0.4
0.2
0.0
0
200
400
600
800
1000

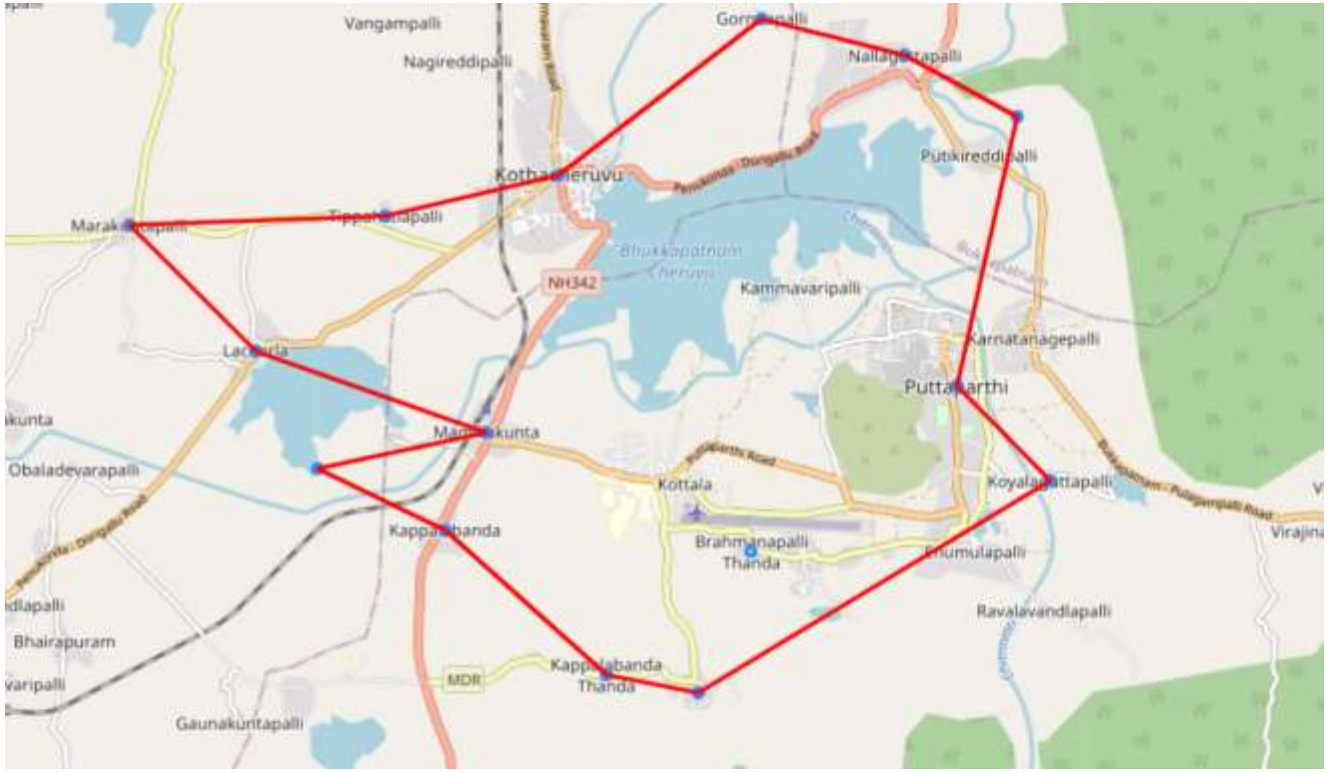


Temperature
Temperature
70
60
50
40
30
20
10
0
0
200
400
600
800
1000



Total length (cost)
Length
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
0
200
400
600
800
1000
Step

Simulated Annealing Results:



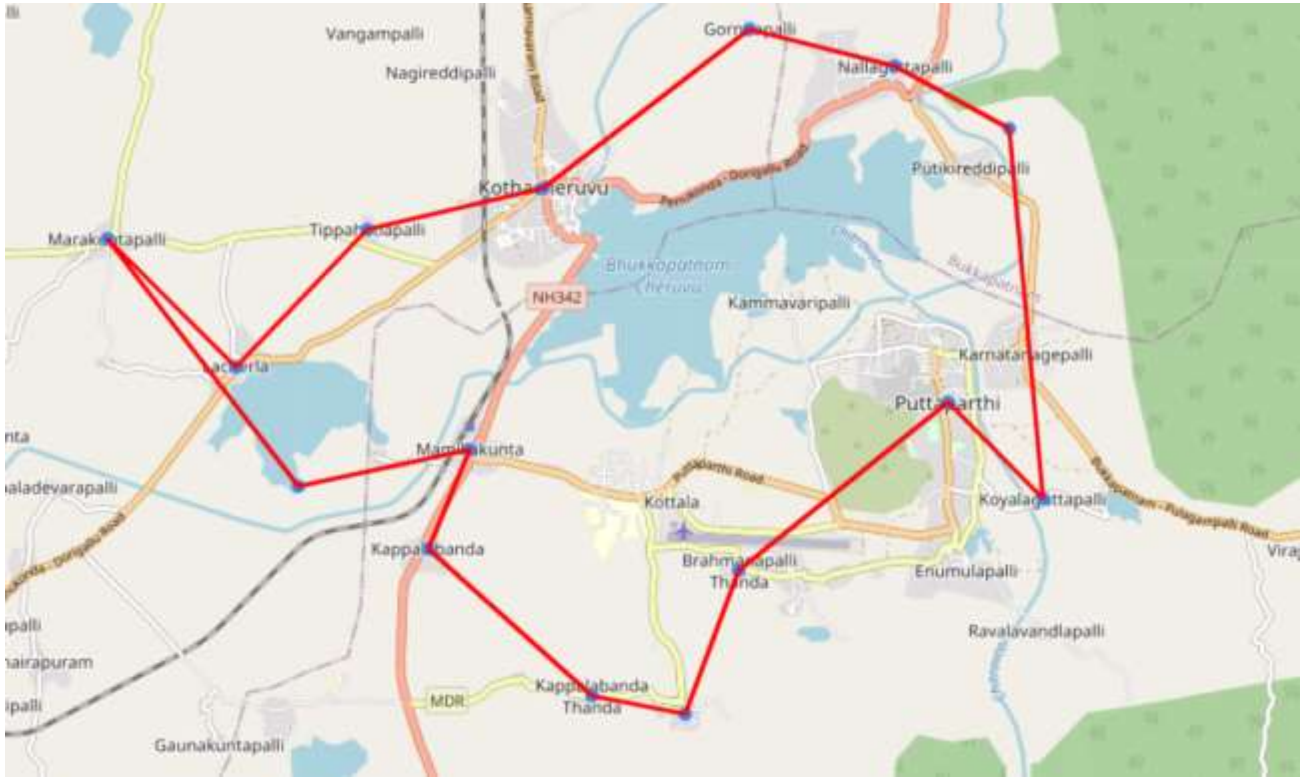

```
Simulated Annealing:
Temperature: 0.9973879496507521, Cooling Rate: 0.003

Simulated Annealing optimal path: [13, 14, 11, 2, 6, 5, 4, 7, 10, 9, 1, 0, 3, 8, 13]
Simulated Annealing total path length: 36.07000000000001
```

Christofides Algorithm:

It guarantees to find a solution that is within a factor of 3/2 of the optimal solution.
Steps -1:
Minimum Spanning Tree (MST): First, a minimum spanning tree of the given graph (representing the cities and distances between them) is constructed.
Step – 2:
Minimum Weight Perfect Matching: Next, the algorithm finds a minimum weight perfect matching of the odd-degree vertices (vertices with an odd number of edges) in the minimum spanning tree.
Step – 3:
Eulerian Circuit: Combining the minimum spanning tree and the matching, an Eulerian circuit (a circuit that visits every edge exactly once) is formed.
Step – 4:
TSP Tour Construction: Finally, the Eulerian circuit is transformed into a TSP tour by skipping already visited vertices.

Christofides Results:




```
tsp_path : [0, 8, 13, 14, 12, 2, 11, 6, 5, 4, 7, 10, 1, 9, 3, 0]
tsp_cost : 35.49
```

Parameter Tuning on ACO:

- The results obtained using the ACO vary very much with that of the lower bound we have found.
- So, by tuning of the parameters involved in the ACO , better results can be obtained.
- The parameters –

  Alpha**(α)**– It accounts for the pheromone level factor in selecting the next step or path.

  Beta(**β)**– It accounts for the distance factor in selecting the next path.

  Rho(**ρ)**– It accounts for the evaporation factor of pheromones released by the ants.

- Methods used:

  a) Bayesian Optimization

  b) Response Surface Methodology

Bayesian Optimization:

- Prior Distribution:

We start by defining prior distributions for the parameters of interest, representing our initial beliefs about their values before observing any data.

- Likelihood Function:

We then collect data by running the ACO algorithm with different parameter configurations and observe its performance metrics, such as tour length or convergence rate.

- Posterior Distribution:

Using Bayes' theorem, we update our beliefs about the parameters based on the observed data, yielding posterior distributions that incorporate both prior knowledge and new evidence.

- Parameter Estimation:

We estimate the parameters' values by summarizing the posterior distributions, such as computing posterior means or medians.

Response Surface Methodology:

- An optimization technique which can be useful when there are multiple variables effecting a process .

Steps:

- Select the parameters to be tuned.
- Define the response i.e. the objective function – distance in the classical TSP
- Use the experimental design techniques to determine the combination of parameters.
- Find the results generated by various set of parameters.
- Create a mathematical model to relate the input parameters to the output response.
- Optimize the

Reinforcement Learning

What is RL and why RL?

- ➢ Type of machine learning algorithm where an agent learns to make decisions by interacting with an environment.
- ➢ It relies on two import terms : action and reward .
- ➢ And it is about taking suitable action to maximize reward in a particular situation.

- Adaptability:
- ➢ RL algorithms can adapt to changes in the environment and offers a more realistic/practical touch.
- ➢ They learn from experience and adjust their strategies accordingly.

- Exploration-Exploitation Balance:
- ➢ RL algorithms naturally balance exploration and exploitation, which can help in escaping local optima.
- ➢ They can explore the search space more efficiently than traditional optimization algorithms.

Q-Learning on traditional TSP:

Initialization:

- Initializes parameters like the number of cities, the number of episodes, learning rate, discount factor, and epsilon for epsilon-greedy policy and also initializes a Q-table with zeros.

Epsilon-Greedy Policy:

- It defines a function for the epsilon-greedy policy, which chooses a random action with probability epsilon, otherwise chooses the action with the maximum Q-value for a given state

The formula for the epsilon-greedy policy is:

$\pi(a|s)$=*random action with probability* ϵ

action with max$a'Q(s,a')$*with probability* $1-\epsilon$

Q-learning Algorithm:

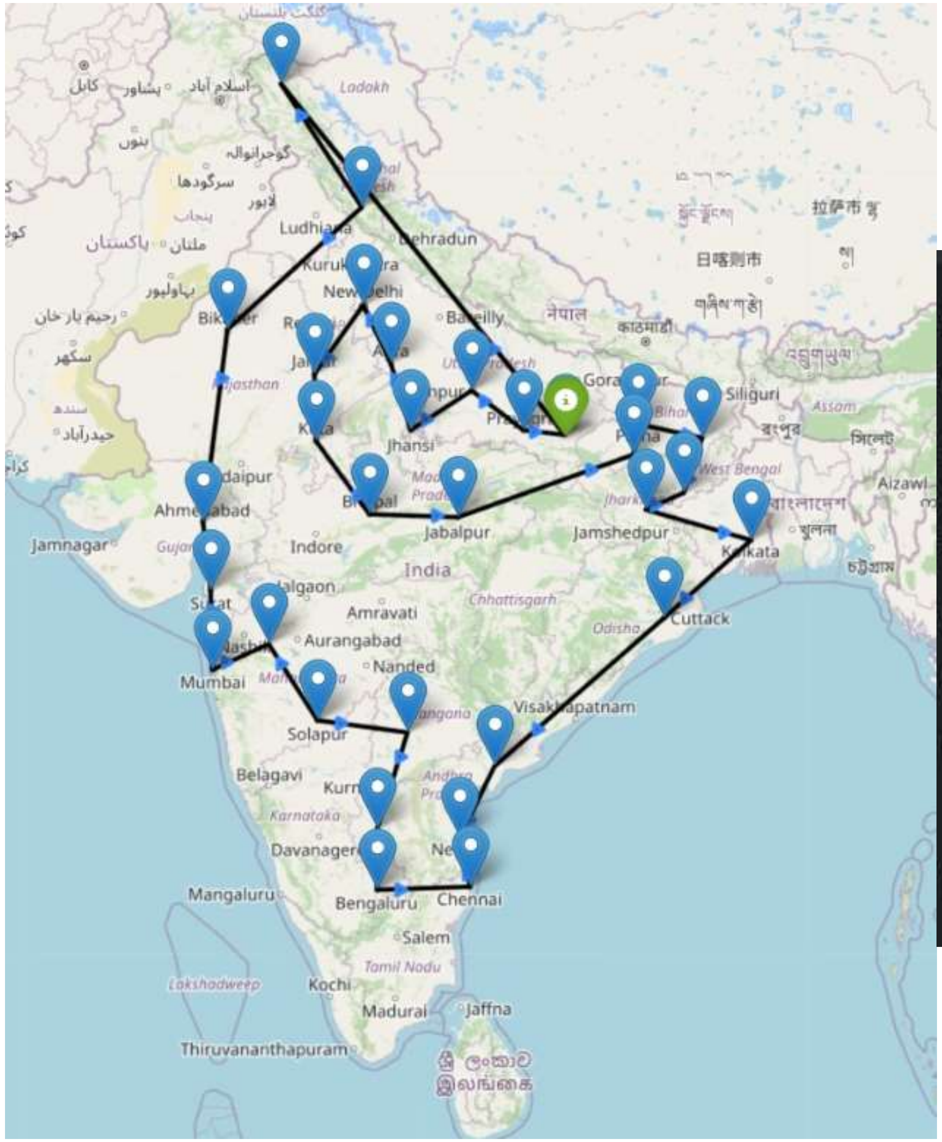
- It runs the Q-learning algorithm for a certain number of episodes.
- In each episode, it starts from a random city, performs a full tour while updating the Q-table based on the rewards obtained, and finally returns to the starting city.

$$Q_{t+1} = Q_t(s,a) + \alpha\left[r(s,a) + \gamma max_{a'} Q(s',a') - Q_t(s,a)\right]$$

Finding Optimal Tour:

- After training, it finds the optimal tour by choosing the action with the maximum Q-value at each step.
- It iterates through all cities to find the shortest tour.

Q-Learning Results:




```
Episode 4995, Total Reward: -29734.0
Episode 4996, Total Reward: -30304.0
Episode 4997, Total Reward: -24780.0
Episode 4998, Total Reward: -30142.0
Episode 4999, Total Reward: -27262.0
Episode 5000, Total Reward: -30298.0
Optimal Tour: [0, 4, 22, 28, 27, 8, 7, 6, 11, 15, 26, 13, 25, 18, 14, 18, 24, 1, 19, 28, 23, 2, 3, 30, 5, 16, 29, 9, 21, 17, 12, 0]
Total Length of Journey: 9397.0

Process finished with exit code 0
```

```
varanasi
gudivada
anantapur
hyderabad
allahabad
Rameswaram
shirdi
kota
jaipur
delhi
ahmedabad
kolkata
bhopal
srinagar
patna
ranchi
jabalpur
mumbai
shimla
dhanbad
nellore
chennai
Bikaner
kanpur
bangalore
cuttack
kochi
```

SARSA Algorithm:

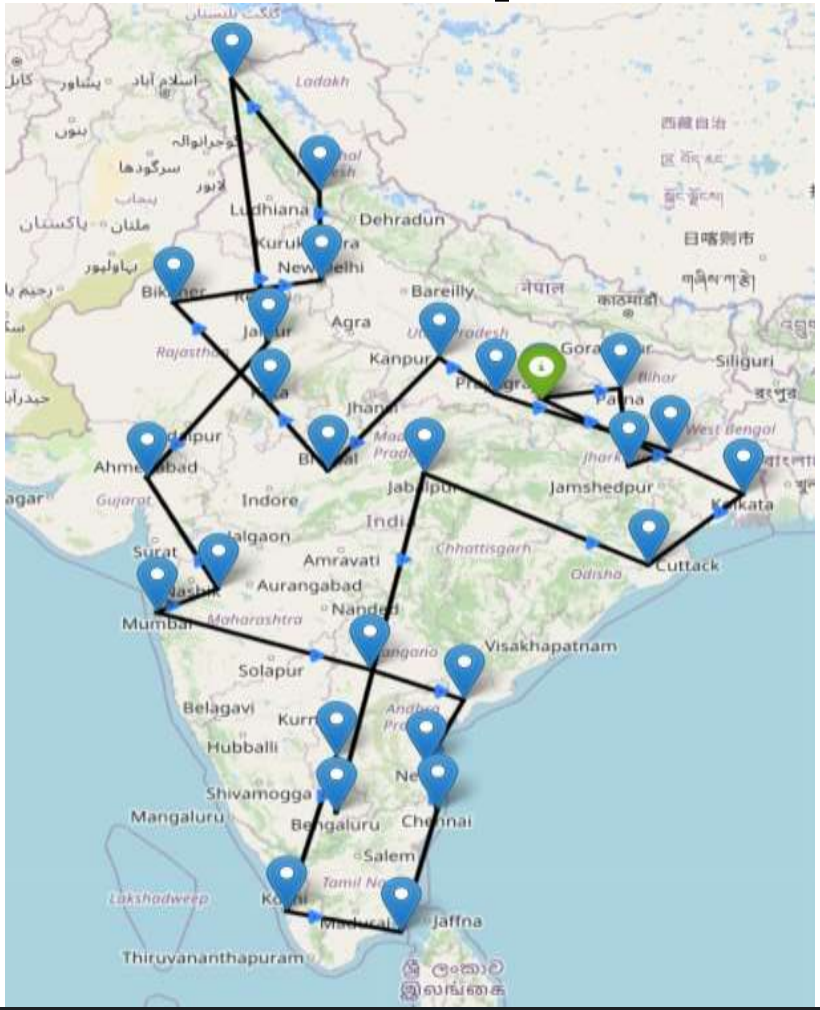
- It is a modified Q learning algorithm where target policy is same as behavior policy.
- The two consecutive state action pairs and the immediate reward received by the agent while transitioning from first state to the next state determine the updated Q value, so this method is called SARSA.

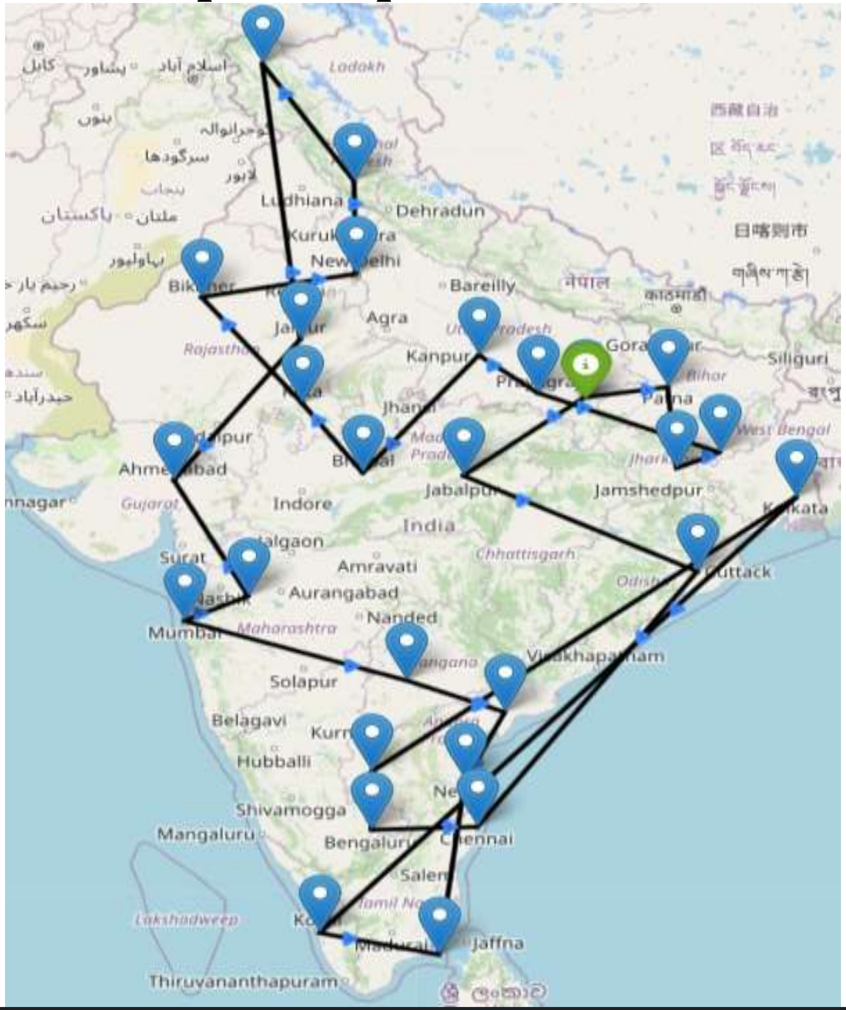
  SARSA : State(s) Action (a) Reward (r) State (s') Action (a').
- As target policy is same as behavior policy , SARSA in an on policy learning algorithm.

$$Q_{t+1} = Q_t(s,a) + \alpha\left[r(s,a) + \gamma Q(s',a') - Q_t(s,a)\right]$$

Comparing Q-Learning and ACO on a Real Life Scenario:

If there is a road block at a particular city in the optimal path - ACO

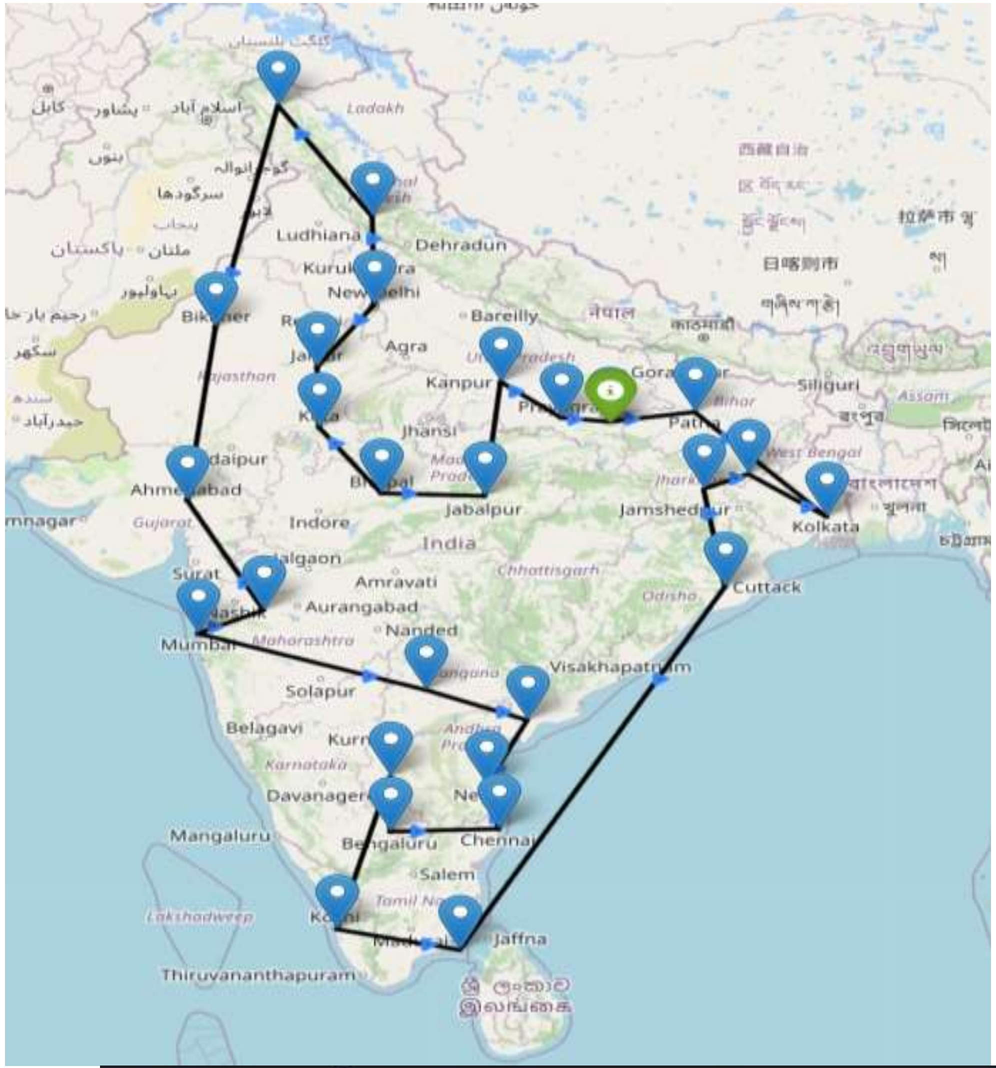


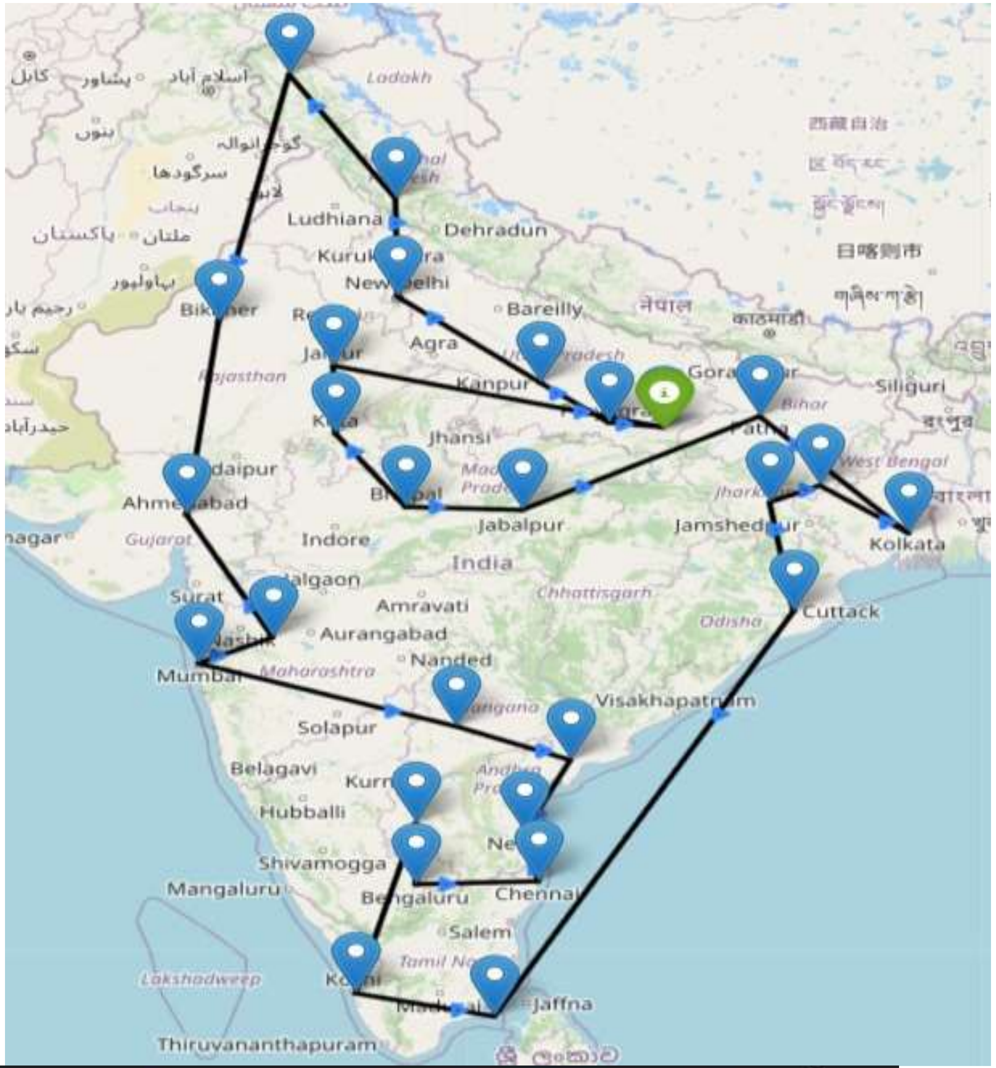



```
varanasi
gudivada
anantapur
hyderabad
allahabad
Rameswaram
shirdi
kota
jaipur
delhi
ahmedabad
kolkata
bhopal
srinagar
patna
ranchi
jabalpur
mumbai
shimla
dhanbad
nellore
chennai
Bikaner
kanpur
bangalore
cuttack
kochi
```

```
11026.0
[0, 14, 15, 19, 4, 23, 12, 7, 22, 9, 18, 13, 8, 10, 6, 17, 3, 1, 20, 21, 5, 26, 2, 24, 16, 25, 11, 0]
Enter x1: 20
Enter x2: 21
13370.0
```

If there is a road block at a particular city in the optimal path – Q-Learning






```
Optimal Tour: [0, 4, 23, 16, 12, 7, 8, 9, 18, 13, 22, 10, 6, 17, 3, 1, 20, 21, 24, 2, 26, 5, 25, 15, 19, 11, 14, 0]
Total Length of Journey: 9620.0
enter x1: 23
enter x2: 16
Optimal Tour: [0, 4, 23, 9, 18, 13, 22, 10, 6, 17, 3, 1, 20, 21, 24, 2, 26, 5, 25, 15, 19, 11, 14, 16, 12, 7, 8, 0]
Total Length of Journey: 10519.0
```

Conclusion

Starting with the natural heuristics like genetic and ACO algorithms, we have explored the Reinforcement learning algorithms like Q-Learning and SARSA for solving the TSP. For solving the traditional TSP , there are many algorithms which gives optimal or near optimal solutions. But , Reinforcement learning offers an advantage over the other algorithms in solving more dynamic scenarios in the environment.

THANK YOU