Reinforcement Learning for the Asymmetric Traveling Salesperson Problem with Precedence Constraints

Elisabeth Bankl

April 3, 2025

Problem Statement 1

- A set of tasks that need to be completed after another
- Some tasks need to be finished before starting others (precedence constraints)
- The execution time of a task depends on the task before it (transition time)
- Objective: Find the best order to execute the tasks that minimizes the total execution time and follows all precedence constraints
- Asymmetric Travelling Salesperson Problem with Precedence Constraints: nodes represent tasks, distances represent transition times

Problem Statement 1

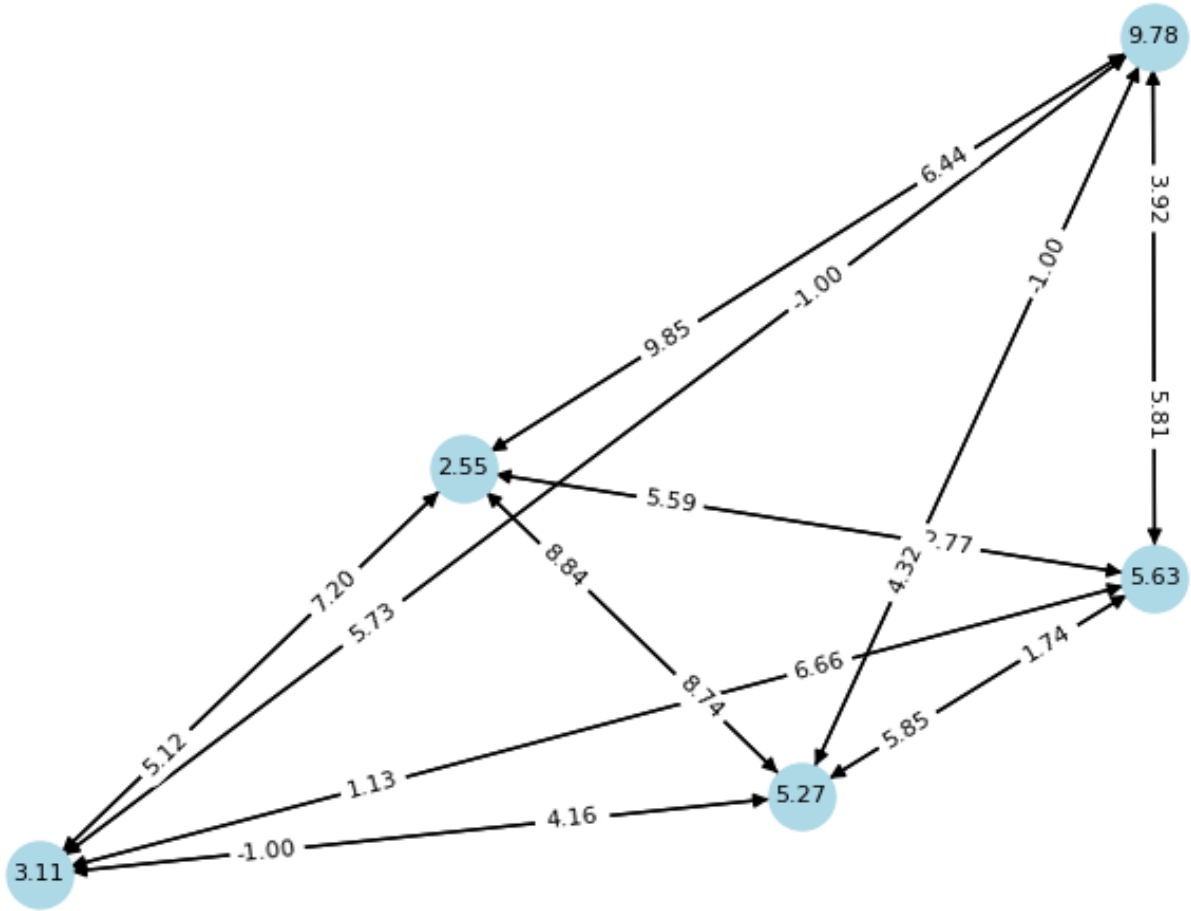

Figure: Graph visualization of the problem

MDP Formulation

- **State Space**: The state includes:
  - **Distance Matrix (D)**: Represents the distances between nodes.
  - **Precedence Matrix (P)**: Indicates which nodes must be visited before others.
  - **Cost Matrix (C)**: Contains the cost of starting at a node and the cost of traveling between nodes.
  - **Visited Nodes Vector (V)**: A binary vector indicating which nodes have been visited.
- **Action Space**: The actions are the nodes that the agent can visit next. However, actions are restricted:
  - Nodes with unfulfilled precedence constraints are not allowed.
  - Nodes that have already been visited are not allowed.
- **Reward Function**: The reward is calculated based on the action taken:
  - If the action violates precedence constraints, a large negative reward is given.
  - If the action is valid, the reward is the negative of the distance between the current node and new node.

Reinforcement Learning Algorithm

- **Algorithm**: Proximal Policy Optimization (PPO)
- **Network Architecture**: MLP

Dataset

- Problem instances randomly generated during training
- Randomly sample distances between nodes, precedence constraints and node costs

Reward Curve for RL Agent

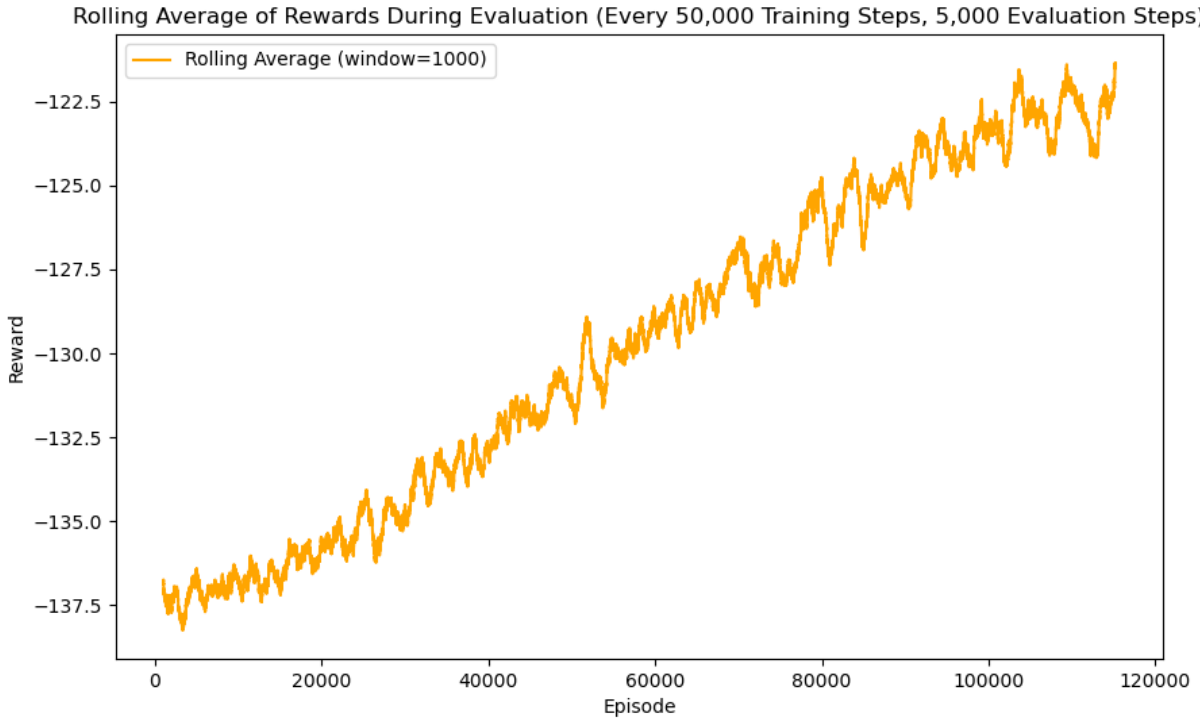

Figure: Reward curve for the reinforcement learning agent during training

Current Limitations

Maximum problem size: 25 nodes

Comparison with simple algorithms

Algorithm	**Average Reward**
Random Algorithm	-137.5
Greedy Algorithm	-85
Reinforcement Learning Agent	-123

Table: Performance comparison of different algorithms