# SMDP Q-Learning and Intra-Option Q-Learning in the Taxi Domain

## 1. Problem Setup

### Available Options

The agent can choose from four predefined **high-level options**, each corresponding to navigating to one of the four fixed locations on the grid:

- **Go to R** – (0, 0)
- **Go to G** – (0, 4)
- **Go to Y** – (4, 0)
- **Go to B** – (4, 3)

Each option internally uses **primitive actions** (north, south, east, west, pickup, dropoff), and the option **terminates** once the taxi reaches the specified location, followed by a pickup or drop-off action depending on the current state. These option policies are trained using Q-Learning.

### State Space Design

- **High-Level Policy State Space**:
  The full Taxi state space (500 states) has been reduced to a **20-state space**, computed as a 5 (passenger locations) × 4 (destination locations) grid. This abstraction is based on the observation that high-level decision-making for the option selection only depends on the passenger's current location and destination—not the taxi's position—thereby reducing complexity and improving learning speed. Although some combinations (i.e. when the passenger and destination are the same) are infeasible in practice—since they would immediately end the episode—they are still included for simplicity and uniformity of implementation.

- **Option Policy State Space**:
  Each option operates over a **25-state space** (5 rows × 5 columns), since the taxi needs to navigate from any position on the grid to a specific location.

# 2. Policies Learned

## SMDP Q-Learning

### High-Level Policy

As seen in Figure 1, the agent selects optimal options based on the decoded passenger and destination locations. The policy follows a logical sequence to minimize total travel time:

**Go to Passenger's Location → Pickup → Go to Destination → Drop Off**

**Example:** If the passenger is at **R** and the destination is **B**:

- Execute **Go to R** → Navigate and pick up the passenger.
- Execute **Go to B** → Navigate and drop off the passenger.

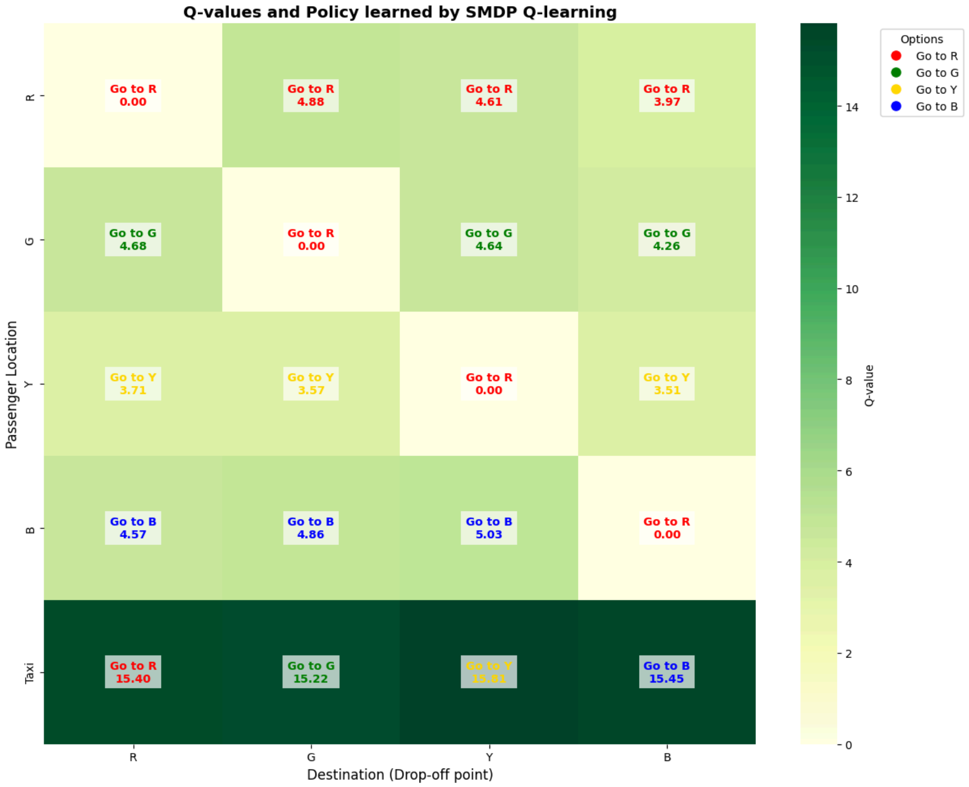

*Figure 1: Maximum Q-value heatmap and option selection policy learned by SMDP Q-learning*

### Option Policies

Each option is responsible for moving the taxi to a fixed location, starting from any grid cell. Once the goal is reached, a pick/drop action is performed based on the state. As shown in Figure 2, Q-plots confirm that each option learns to reach its target efficiently by minimizing steps.

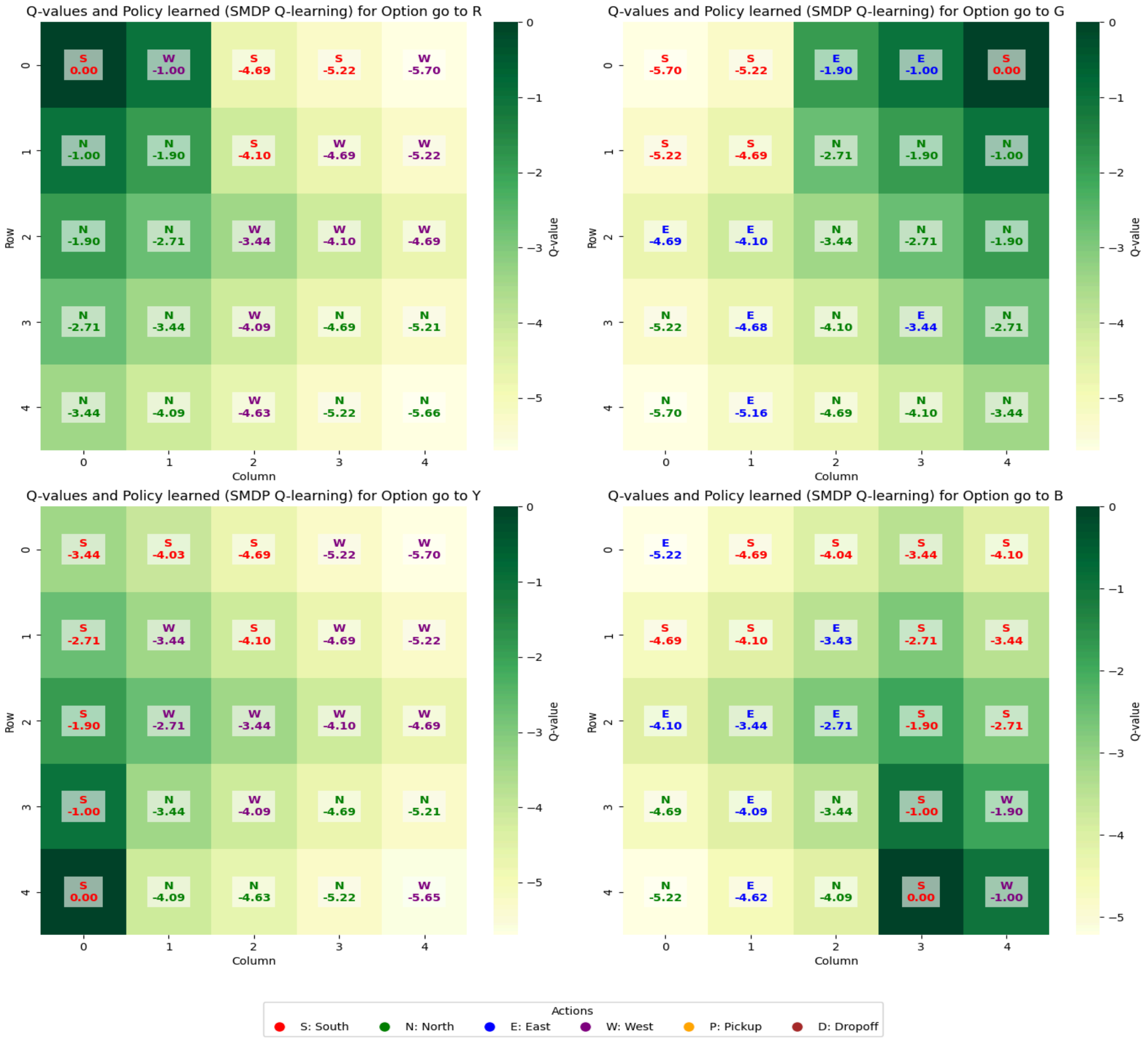

*Figure 2: Maximum Q-value and policy learned for SMDP Q-learning's options*

## Intra-Option Q-Learning

### High-Level Policy

From Figure 3, we observe that the intra-option policy mirrors the SMDP approach in structure and goal. It also minimizes time to goal by executing:

**Go to Passenger's Location → Pickup → Go to Destination → Drop Off**

**Example:** If the passenger is at **Y** and the destination is **G**:

- Execute **Go to Y** → Navigate and pick up.
- Execute **Go to G** → Navigate and drop off.

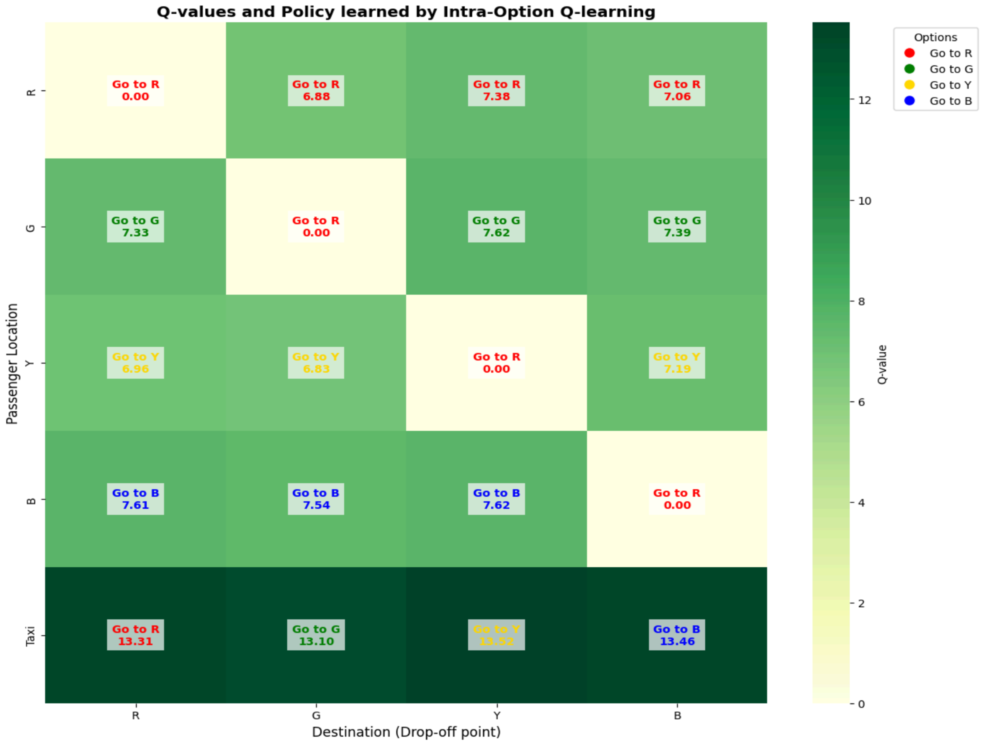

*Figure 3: Maximum Q-value heatmap and option selection policy learned by Intra-Option Q-learning*

## Option Policies

Option behaviors are the same as in SMDP: navigate to the goal location and terminate with a pick/drop. As illustrated in Figure 4, the Q-plots confirm that each option effectively learns an optimal path, minimizing the number of steps required to reach its target location.

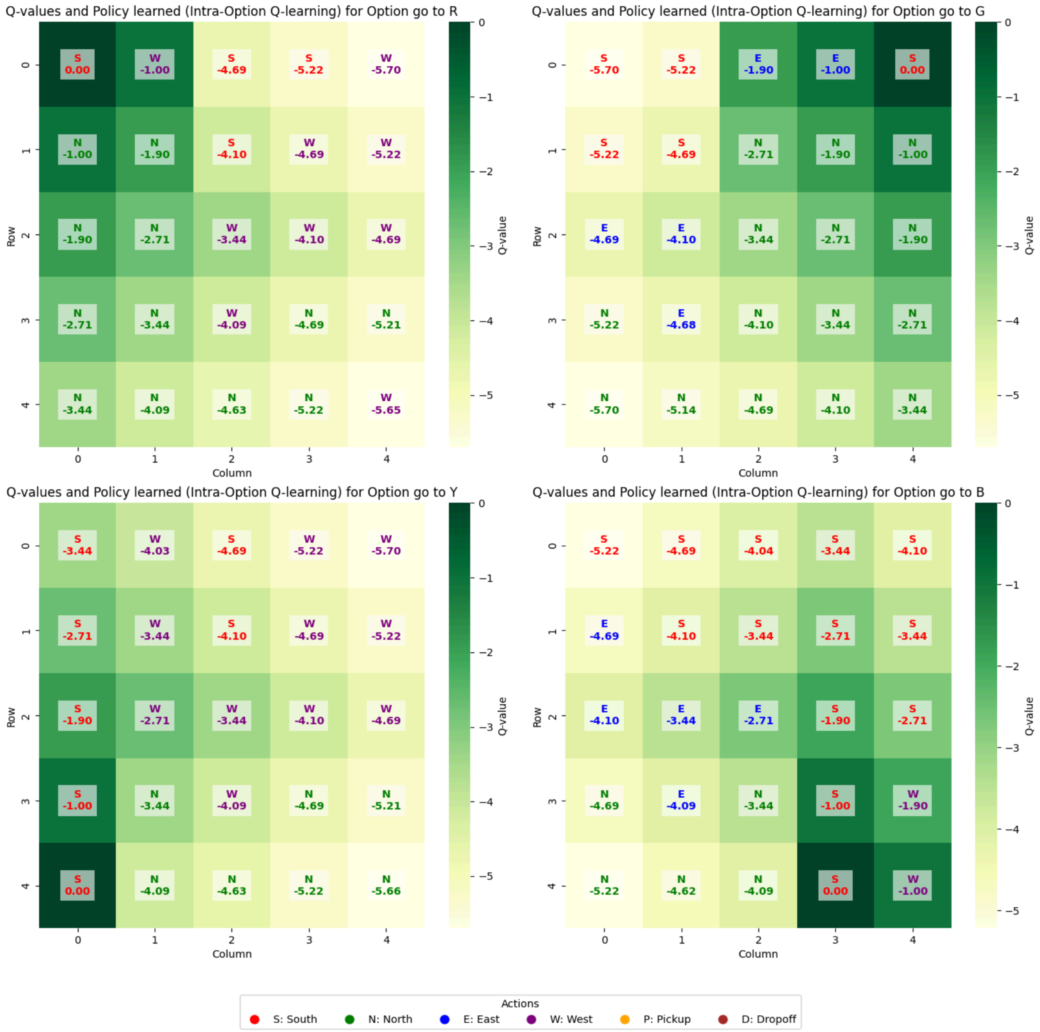

*Figure 4: Maximum Q-value and policy learned for Intra-Option Q-learning's options*

## Why These Policies Are Learned

Both SMDP and Intra-Option Q-Learning converge on the same **efficient sequence of options** for the task. This is because:

- Choosing a wrong option leads to extra steps and delay, as an option only terminates when its goal is reached.
- Longer execution times within options lead to more negative rewards due to step penalties.
- Over time, both methods refine each option's internal policy to minimize steps to goal and learn to sequence options optimally for end-to-end task completion.

# 3. Alternate Option Set

As a comparison, a different set of options is introduced:

- **Pick up Passenger**
- **Drop off Passenger**

These options are mutually exclusive from the location-specific options above. The **Pick-up** option terminates after the taxi reaches the passenger and performs the pickup. The **Drop-off** option terminates after reaching the destination and executing drop off.

## State Space

- **High-Level Policy**: Maintains the same reduced 20-state space (passenger location × destination location).
- **Option Policies**:
  These have a larger state space of **5 × 5 × 5 = 125 states** (grid position × passenger location). This is required because:
  - The pickup option needs to know the taxi's and passenger's locations.
  - The drop-off option needs to know the taxi's location and the destination.

*Note:* The number of passenger locations is used to define the third dimension of the state space, even for the drop-off option. This choice ensures consistent state dimensions across both options and simplifies the implementation of the learning algorithms.

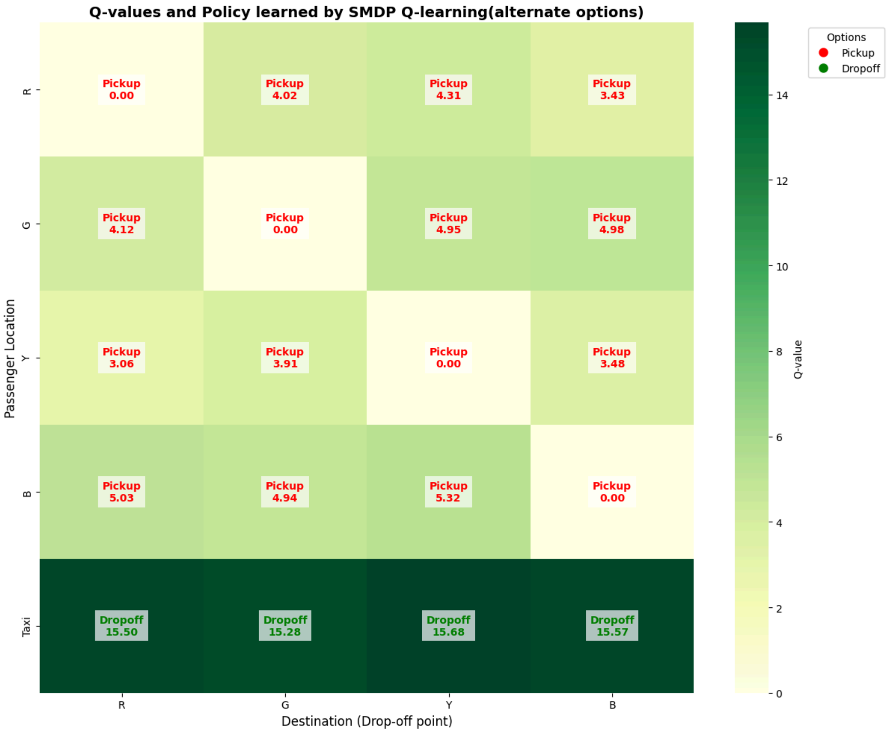

*Figure 5: Maximum Q-value and policy learned by SMDP Q-learning with alternate options*

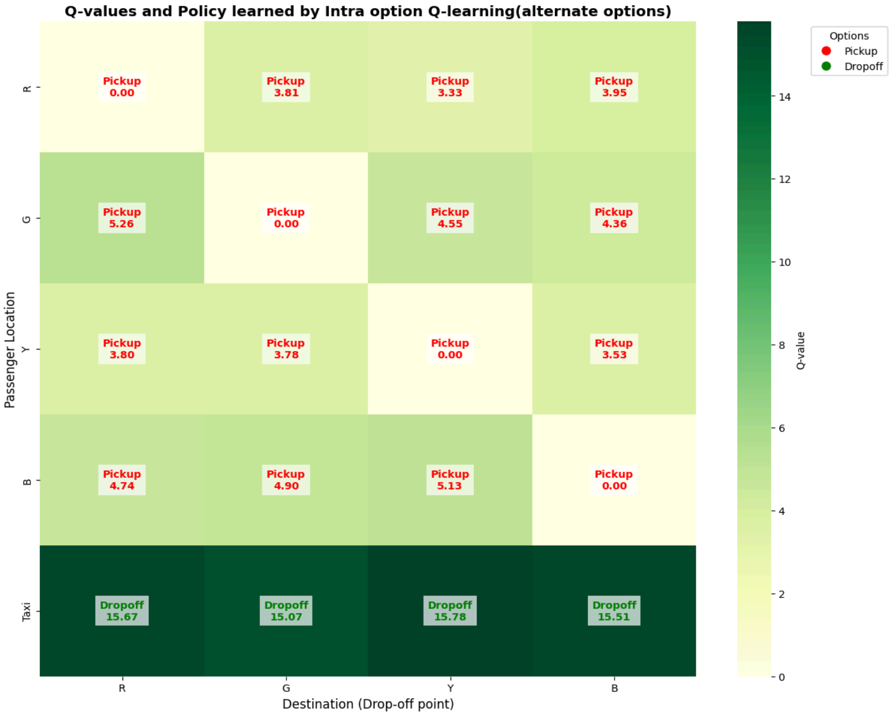

*Figure 6: Maximum Q-value and policy learned by Intra-Option Q-learning with alternate options*

## Observations

- Both SMDP (Figure 5) and Intra-Option (Figure 6) approaches with alternate options learn the correct high-level behavior:  
**Pick up when the passenger is not inside the taxi, and drop off when they are.**

- **Early Training**: These options suffer from **higher negative rewards** (Figure 7) and **step counts** (Figure 8). This is likely due to the larger state space of each option, making it harder for the agent to learn optimal navigation to the passenger/destination early on.

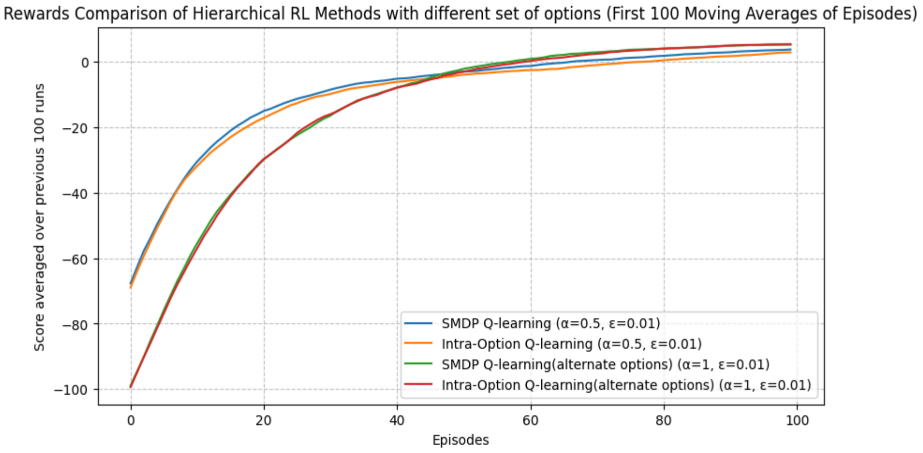

*Figure 7: Initial moving average (over 100 episodes) of rewards for different methods*

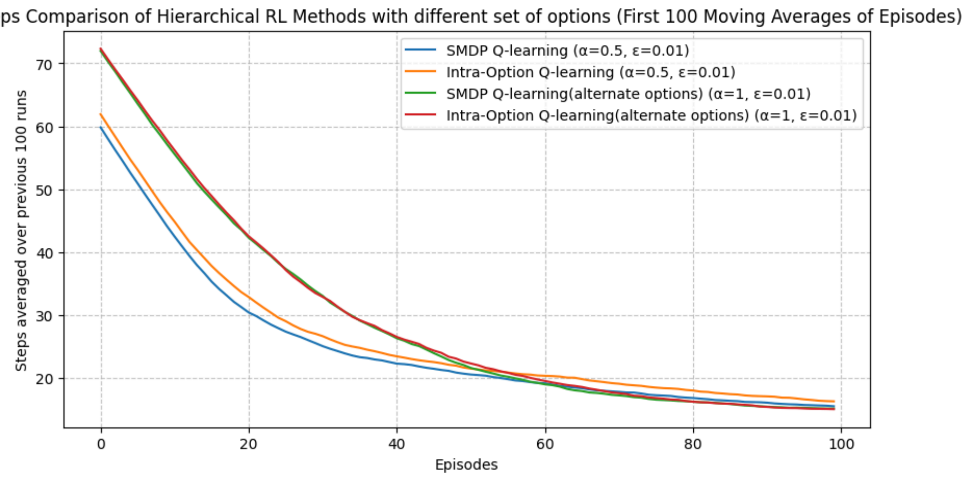

*Figure 8: Initial moving average (over 100 episodes) of step counts for different methods*

- Performance improves and even **briefly surpasses** the given location-based options, before both approaches **converge to similar levels** of performance, as shown in Figure 9 (rewards) and Figure 10 (step counts per episode). This convergence indicates that the algorithms in both the cases ultimately learn an efficient high-level option selection policy along with well-optimized option policies for the task.

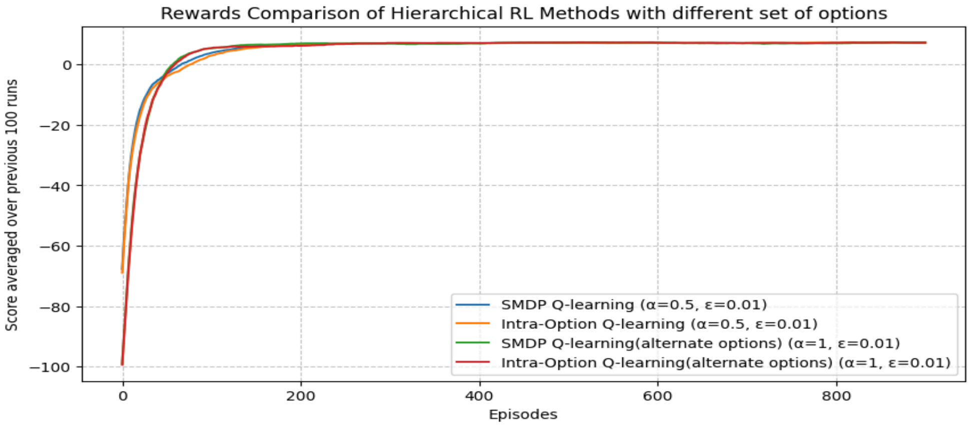

*Figure 9: Moving average (over 100 episodes) of rewards for different methods*

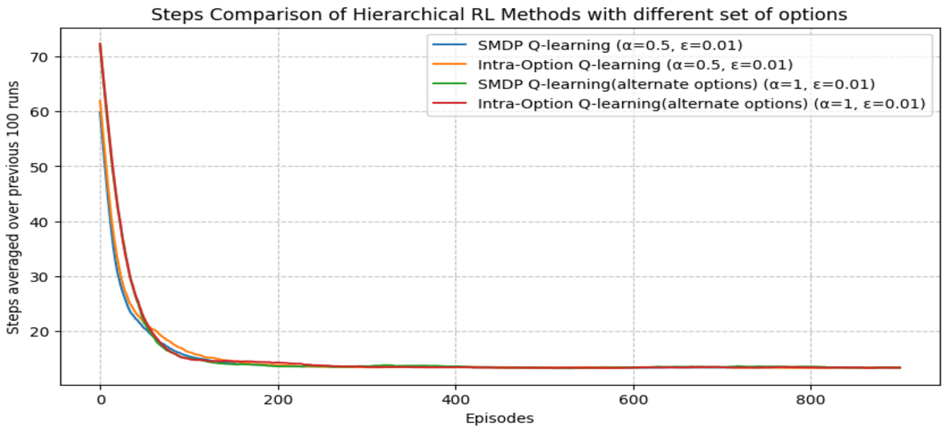

*Figure 10: Moving average (over 100 episodes) of step counts for different methods*

# 4. SMDP vs. Intra-Option Q-Learning

| **Feature** | **SMDP Q-Learning** | **Intra-Option Q-Learning** |
|---|---|---|
| **Update Frequency** | One update at the end of each option | Updates on every primitive step |
| **Update Scope** | Only the executing state-option pair | All options consistent with the action chosen by the current option. |
| **Reward Aggregation** | Cumulative discounted reward (multi-step) | Immediate rewards at each step |
| **Option Type** | Semi-Markov (treats Markov and Semi-Markov similarly) | Markov only |
| **Learning Efficiency** | Slower but stable | Faster, potentially less stable |

## Insights from Learning Curves and Q-value plots

- From Figures 9 and 10, both algorithms show **similar learning curves** for both the option approaches, indicating **no clear advantage** in convergence rate in this setup. This may be due to the small number of options, reducing the benefit of intra-option updates.

- However, comparing Figure 1 vs. Figure 3, the maximum Q-values in intra-option Q-learning are **higher** for most states (except when the passenger is in the taxi). This likely results from more frequent updates to the Q-values of state-option pairs.

- This improvement is **not observed in the alternate options** (Figures 5 and 6), possibly because there are **only two options**, limiting the benefit of frequent updates in intra-option.