Reinforcement Learning in Cancer Research: Optimizing the p53-Mdm2 Feedback Loop

Indrajeet Roy
Department of Electrical and Computer Engineering, Northeastern University
roy.i@northeastern.edu

Introduction

MDP for the p53-Mdm2 negative feedback loop network consists of 4 components/genes, represented in the state vector $\mathbf{s}_k = [\mathrm{ATM}, \mathrm{p53}, \mathrm{Wpi}, \mathrm{MDM2}]^T$. At any time, the state value of each gene can be: 0 (OFF) and 1 (ON).

$$\mathcal{S} = \left\{[0,0,0,0]^T, [0,0,0,1]^T, [0,0,1,0]^T, [0,0,1,1]^T, \ldots, [1,1,1,0]^T, [1,1,1,1]^T\right\}$$
$$\mathcal{A} = \left\{[0,0,0,0]^T, [1,0,0,0]^T, [0,1,0,0]^T, [0,0,1,0]^T, [0,0,0,1]^T\right\}$$

The state transition probability equation is defined as:

$$\mathbf{s}_k = \mathbf{C} \cdot (\mathbf{s}_{k-1} \oplus \mathbf{a}_{k-1}) \oplus \mathbf{n}_k$$

Where:

- $\mathbf{s}_k$ is the state vector at time k.
- $\mathbf{C}$ is the connectivity matrix that defines the relationship between states.
- $\mathbf{a}_{k-1}$ is the action vector applied at time $k-1$ affecting the state transition.
- $\mathbf{n}_k$ is the state transition noise vector that introduces randomness into the transition.
- $\oplus$ is the XOR operation used for combining the state vector and action vector as well as adding the noise vector.

The transition matrices, representing the probability of moving from any state to other states under different control inputs is defined as:

$$(M(\mathbf{a}))_{ij} = p^{\|\mathbf{s}_i - (\mathbf{C}\mathbf{s}_i \oplus \mathbf{a})\|_1} \cdot (1-p)^{4 - \|\mathbf{s}_i - (\mathbf{C}\mathbf{s}_i \oplus \mathbf{a})\|_1}$$

where $\|\mathbf{v}\|_1 = \sum_i |\mathbf{v}(i)|$ is the 1-norm, summing the absolute values of the elements of the vector $\mathbf{v}$.

The reward function $R(\mathbf{s}, \mathbf{a}, \mathbf{s}')$ is defined as:

$$R(\mathbf{s}, \mathbf{a}, \mathbf{s}') = 5s'(1) + 5s'(2) + 5s'(3) + 5s'(4) - |\mathbf{a}|$$

where $|\mathbf{a}|$ sums the absolute value of the elements of the action vector $\mathbf{a}$. The activation of each gene contributes a reward of $+5$, and actions $\mathbf{a}_2$ to $\mathbf{a}_5$ incur a cost of -1.

Model Based Algorithm Results

Value Iteration

Matrix-form Value Iteration for discount factor $\gamma = 0.95$, convergence threshold $\theta = 0.01$, and noise parameter $p = 0.05$

Optimal policy π^* : $[2,2,2,2,2,2,2,2,3,2,2,2,4,2,2,2]$

Average activation of genes, AvgA over 100 episodes under the obtained optimal policy: 2.84420

Average activation of genes, AvgA over 100 episodes under the no control policy: 0.49355

Number of Iteration Steps: 142
Matrix-form Value Iteration for discount factor $\gamma = 0.95$, convergence threshold $\theta = 0.01$, and noise parameter $p = 0.2$

Optimal policy π^* : $[2, 2, 2, 2, 2, 2, 2, 2, 3, 2, 2, 2, 4, 2, 2, 2]$

Average activation of genes, AvgA over 100 episodes under the obtained optimal policy: 2.3942

Average activation of genes, AvgA over 100 episodes under the no control policy: 1.2607

Number of Iteration Steps: 138

Matrix-form Value Iteration for discount factor $\gamma = 0.95$, convergence threshold $\theta = 0.01$, and noise parameter $p = 0.45$

Optimal policy π^* : $[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]$

Average activation of genes, AvgA over 100 episodes under the obtained optimal policy: 1.90165

Average activation of genes, AvgA over 100 episodes under the no control policy: 1.88935

Number of Iteration Steps: 135
Matrix-form Policy Iteration for discount factor $\gamma = 0.95$, convergence threshold $\theta = 0.01$, and noise parameter $p = 0.05$

Optimal policy π^* : $[2, 2, 2, 2, 2, 2, 2, 2, 3, 2, 2, 2, 4, 2, 2, 2]$

Average activation of genes, AvgA over 100 episodes under the obtained optimal policy: 2.3942

Average activation of genes, AvgA over 100 episodes under the no control policy: 1.2607

Number of Iteration Steps: 3

Policy Iteration

Matrix-form Policy Iteration for discount factor $\gamma = 0.95$, convergence threshold $\theta = 0.01$, and noise parameter $p = 0.2$

Optimal policy π^* : $[2, 2, 2, 2, 2, 2, 2, 2, 3, 2, 2, 2, 4, 2, 2, 2]$

Average activation of genes, AvgA over 100 episodes under the obtained optimal policy: 2.3942

Average activation of genes, AvgA over 100 episodes under the no control policy: 1.2607

Number of Iteration Steps: 3

Matrix-form Policy Iteration for discount factor $\gamma = 0.95$, convergence threshold $\theta = 0.01$, and noise parameter $p = 0.45$

Optimal policy π^* : $[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]$

Average activation of genes, AvgA over 100 episodes under the obtained optimal policy: 1.9138499999999998

Average activation of genes, AvgA over 100 episodes under the no control policy: 1.9044500000000002

Number of Iteration Steps: 1

Analysis

The optimal policy is sensitive to the level of noise in the system.At lower noise levels ($p = 0.05$ and $p = 0.2$), the system's response to control actions is more predictable and consistent.The Optimal policy for lower noise levels suggesting control actions is effective in controlling/perturbing the system towards the objective all active component state ([1,1,1,1]).The consistency of the optimal policy from ($p = 0.05$) to ($p = 0.2$) highlights the strategy's robustness, effectively maintaining its effectiveness despite an increase in noise.

For $p = 0.05$, the AvgA under the optimal policy is significantly higher in comparison to the AvgA under the no control policy, indicating that system is more controllable by strategic actions, increasing gene activation rates. The higher AvgA also suggests that the benefits of control actions are higher in comparison to the cost of implementing the control actions. The AvgA computed for $p = 0.2$) decreases in comparison to $p = 0.05$, both under the optimal policy and no control policy, indicating that the increased noise negatively impacts the system's ability to sustain high levels of gene activation and controllability.

At higher noise levels ($p = 0.45$), the effectiveness of control actions reduces due to increased unpredictability in the system's behavior and the optimal policy shifts towards not taking any action. The noise introduces significant variability to the system that the effect of any control action is reduced, making it impractical to attempt any control action.

The AvgA computed for ($p = 0.45$) decreases in comparison to ($p = 0.05$ and $p = 0.2$) and is nearly the same under both the optimal policy and no control policy highlighting the reduced effectiveness of control actions.At high noise levels, control becomes less effective, and the system's natural dynamics influence the system behavior more.

The obtained policies are the same across all tested noise levels for both policy iteration and value iteration.Both policy iteration and value iteration effectively identify the same optimal policies for controlling the gene regulatory network under various noise conditions but policy iteration highlights an advantage in terms of convergence speed, requiring fewer iteration steps to compute the optimal policy.

Model Based Algorithm Results

Q-Learning

a2	a2	a2	a2	a2	a2	a2	a2	a1	a2	a1	a2	a1	a2	a1	a2
a2	a2	a2	a2	a2	a2	a2	a2	a1	a3	a1	a2	a1	a2	a2	a2
a2	a2	a2	a2	a2	a2	a2	a2	a3	a2	a1	a2	a1	a2	a2	a2
a2	a2	a2	a2	a2	a2	a2	a2	a1	a1	a2	a2	a1	a2	a1	a2
a2	a2	a2	a2	a2	a2	a2	a2	a1	a1	a1	a2	a1	a2	a2	a2
a2	a2	a2	a2	a2	a2	a2	a2	a1	a2	a1	a2	a1	a2	a1	a2
a2	a2	a2	a2	a2	a2	a2	a2	a1	a1	a1	a2	a1	a2	a1	a2
a2	a2	a2	a2	a2	a2	a2	a2	a1	a2	a1	a2	a1	a2	a1	a2
a2	a2	a2	a2	a2	a2	a2	a2	a1	a1	a1	a2	a1	a2	a2	a2
a2	a2	a2	a2	a2	a2	a2	a2	a1	a2	a1	a2	a1	a2	a2	a2

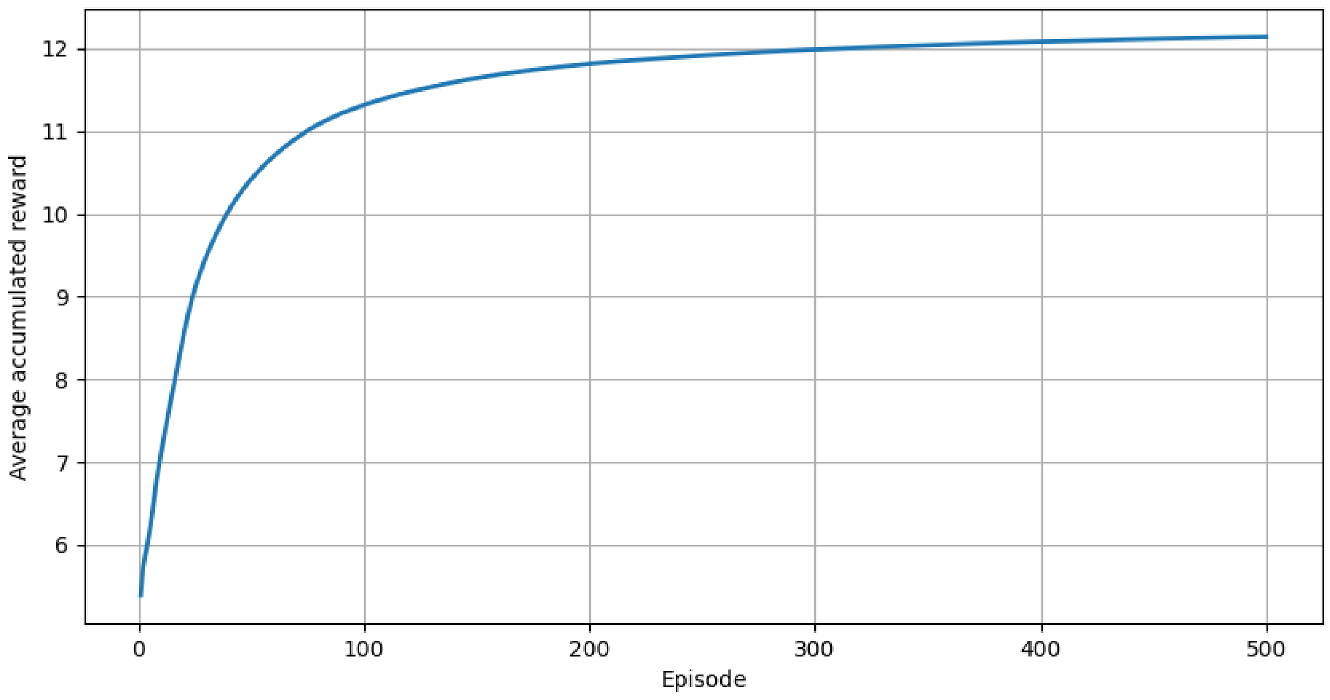

Figure 1: Average accumulated reward (in 10 independent runs) w.r.t episode number for Q-Learning

SARSA

a2	a2	a2	a2	a2	a2	a2	a2	a3	a3	a1	a2	a1	a2	a2	a2
a2	a3	a2	a2	a2	a2	a2	a2	a1	a2	a2	a3	a1	a2	a2	a2
a2	a2	a2	a2	a2	a2	a2	a2	a1	a2	a3	a2	a1	a2	a2	a2
a2	a2	a2	a2	a2	a2	a2	a2	a1	a2	a2	a2	a1	a2	a2	a2
a2	a2	a2	a2	a2	a2	a2	a2	a1	a3	a1	a2	a1	a1	a2	a2
a2	a2	a2	a2	a2	a2	a2	a2	a1	a3	a1	a2	a1	a2	a2	a2
a2	a3	a2	a2	a2	a2	a2	a2	a1	a2	a1	a2	a1	a2	a2	a2
a2	a2	a2	a2	a2	a2	a2	a2	a1	a2	a1	a2	a1	a2	a2	a2
a2	a2	a2	a2	a2	a2	a2	a2	a1	a3	a1	a2	a1	a2	a2	a2
a2	a2	a2	a2	a2	a2	a2	a2	a1	a3	a1	a2	a1	a2	a1	a2

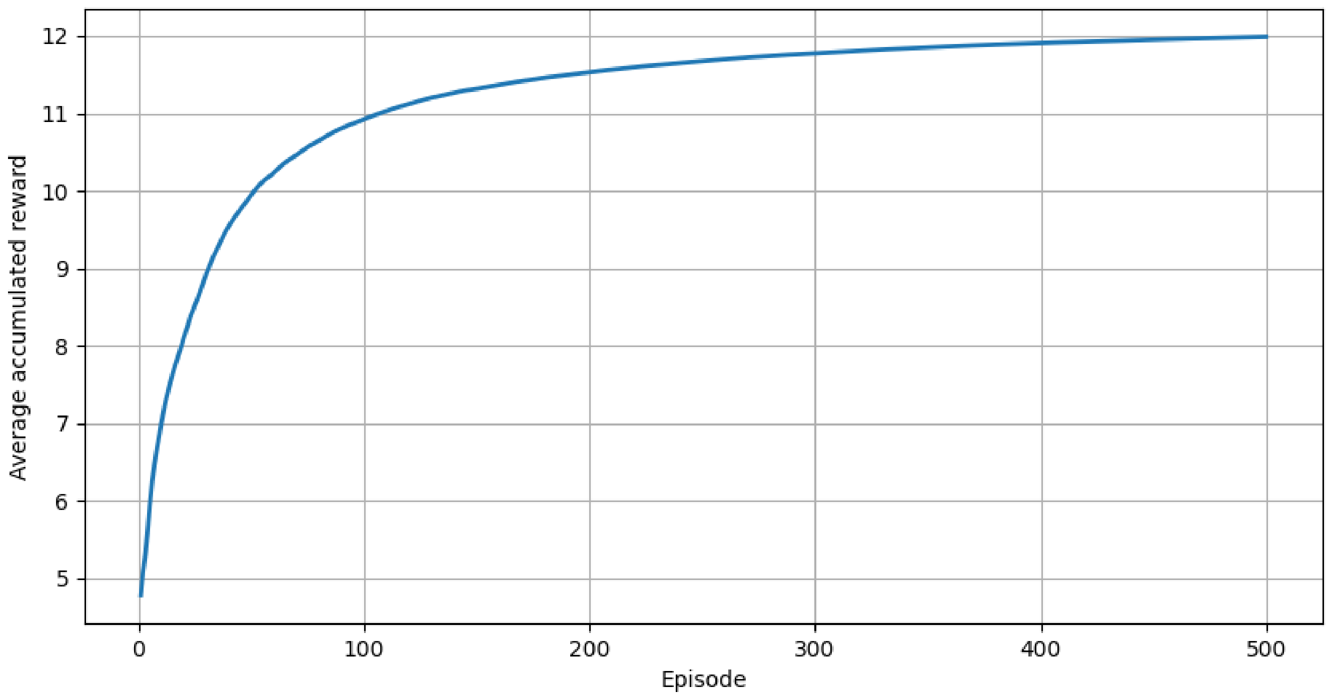

Figure 2: Average accumulated reward (in 10 independent runs) w.r.t episode number for SARSA

SARSA-lambda (λ)

a2	a2	a1	a2	a2	a2	a2	a1	a3	a2	a2	a2	a1	a2	a2	a2
a3	a3	a3	a2	a1	a2	a2	a1	a3	a2	a2	a3	a1	a2	a1	a2
a2	a2	a3	a2	a2	a2	a1	a1	a1	a2	a3	a2	a1	a1	a1	a2
a2	a3	a2	a2	a2	a2	a2	a1	a1	a2	a3	a3	a1	a2	a1	a2
a3	a3	a2	a2	a1	a2	a2	a1	a1	a3	a2	a2	a2	a2	a1	a2
a2	a1	a3	a3	a2	a2	a2	a1	a1	a2	a2	a2	a1	a2	a1	a1
a3	a2	a2	a2	a1	a2	a1	a1	a1	a2	a3	a3	a1	a2	a1	a2
a3	a3	a2	a2	a2	a2	a2	a3	a1	a2	a2	a2	a1	a2	a2	a2
a2	a1	a2	a2	a2	a1	a1	a1	a1	a3	a3	a2	a3	a1	a2	a1
a2	a3	a2	a3	a2	a2	a2	a3	a1	a2	a3	a1	a1	a2	a1	a1

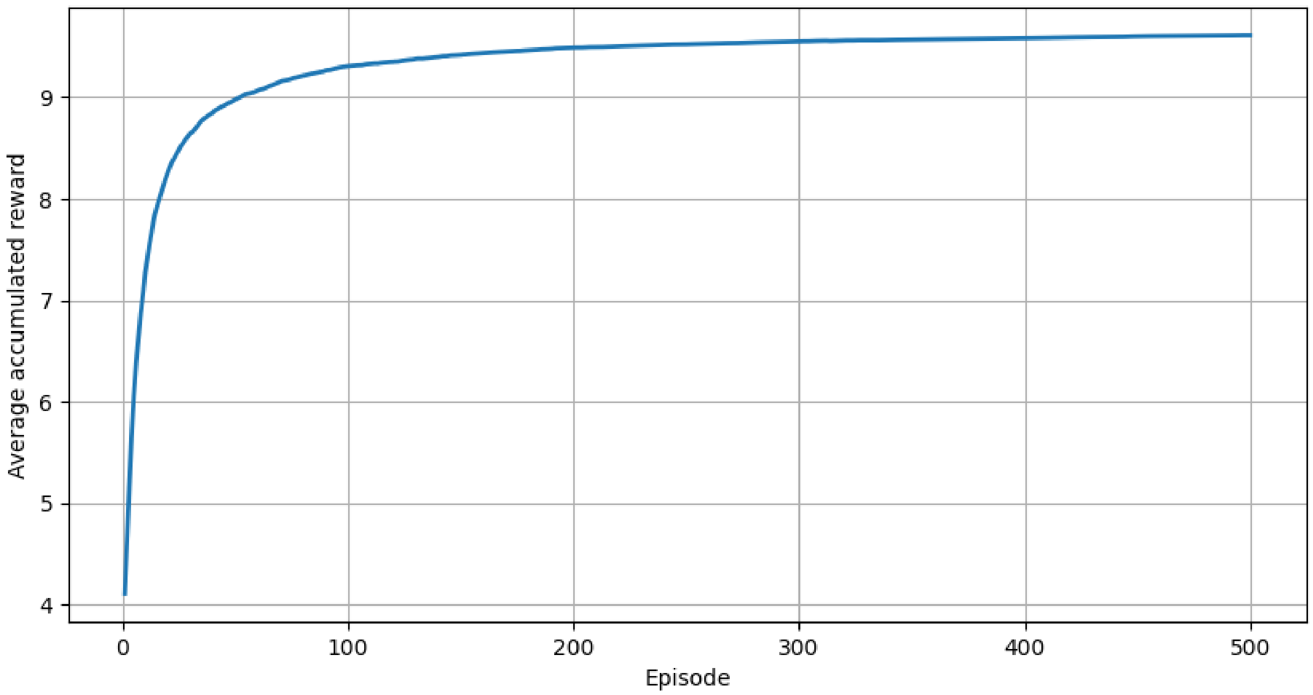
Figure 3: Average accumulated reward (in 10 independent runs) w.r.t episode number for SARSA (λ)

Actor-Critic

a2	a2	a3	a3	a2	a2	a2	a2	a1	a2	a2	a2	a3	a1	a2	a2
a3	a2	a3	a2	a2	a2	a2	a2	a3	a2	a2	a2	a1	a2	a1	a2
a2	a2	a3	a2	a2	a2	a2	a2	a3	a3	a3	a2	a3	a2	a2	a2
a3	a3	a2	a2	a2	a2	a2	a2	a3	a3	a1	a3	a1	a2	a1	a2
a2	a3	a3	a3	a2	a2	a2	a2	a2	a2	a3	a2	a3	a2	a2	a2
a2	a2	a3	a2	a2	a2	a2	a2	a1	a1	a3	a2	a3	a1	a2	a2
a2	a2	a2	a2	a2	a2	a2	a2	a1	a3	a1	a2	a3	a2	a1	a2
a2	a3	a2	a2	a2	a2	a2	a2	a3	a3	a2	a2	a3	a2	a2	a2
a3	a2	a2	a2	a2	a2	a2	a2	a3	a3	a3	a2	a1	a2	a2	a2
a2	a3	a2	a2	a2	a2	a2	a2	a2	a2	a3	a2	a1	a2	a2	a2

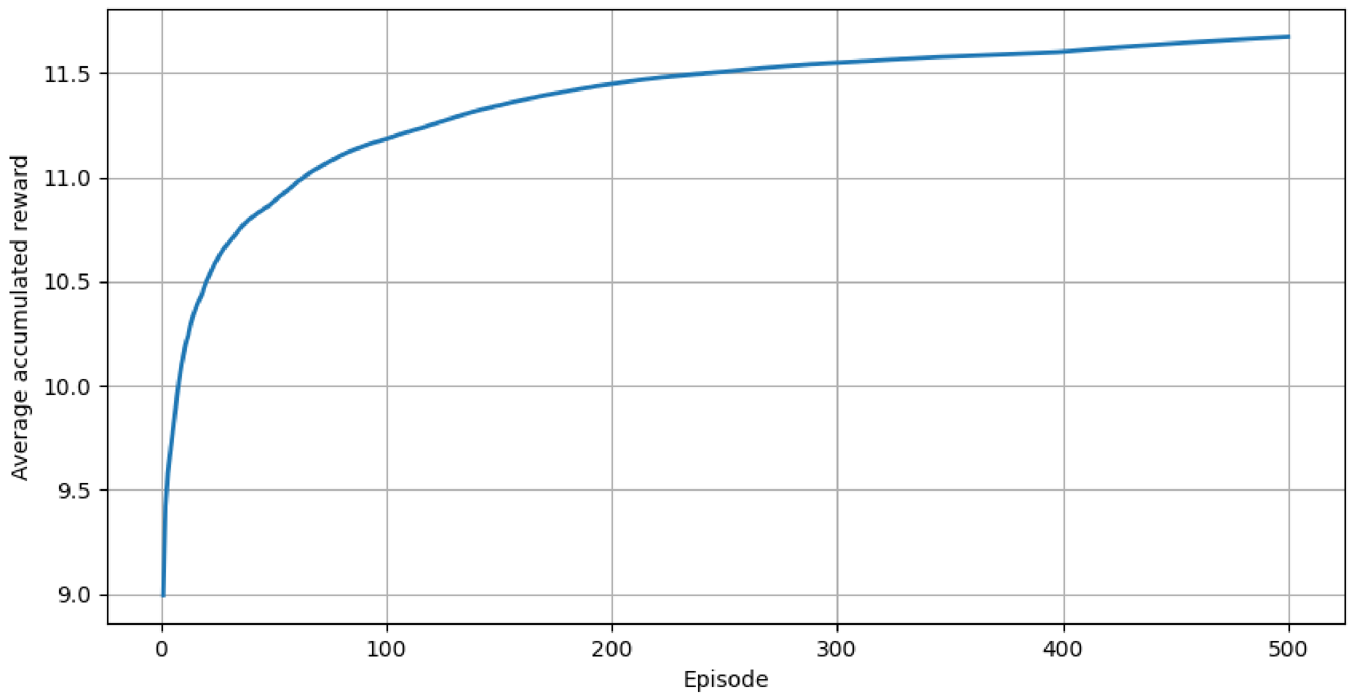

Figure 4: Average accumulated reward (in 10 independent runs) w.r.t episode number for Actor-Critic

Analysis

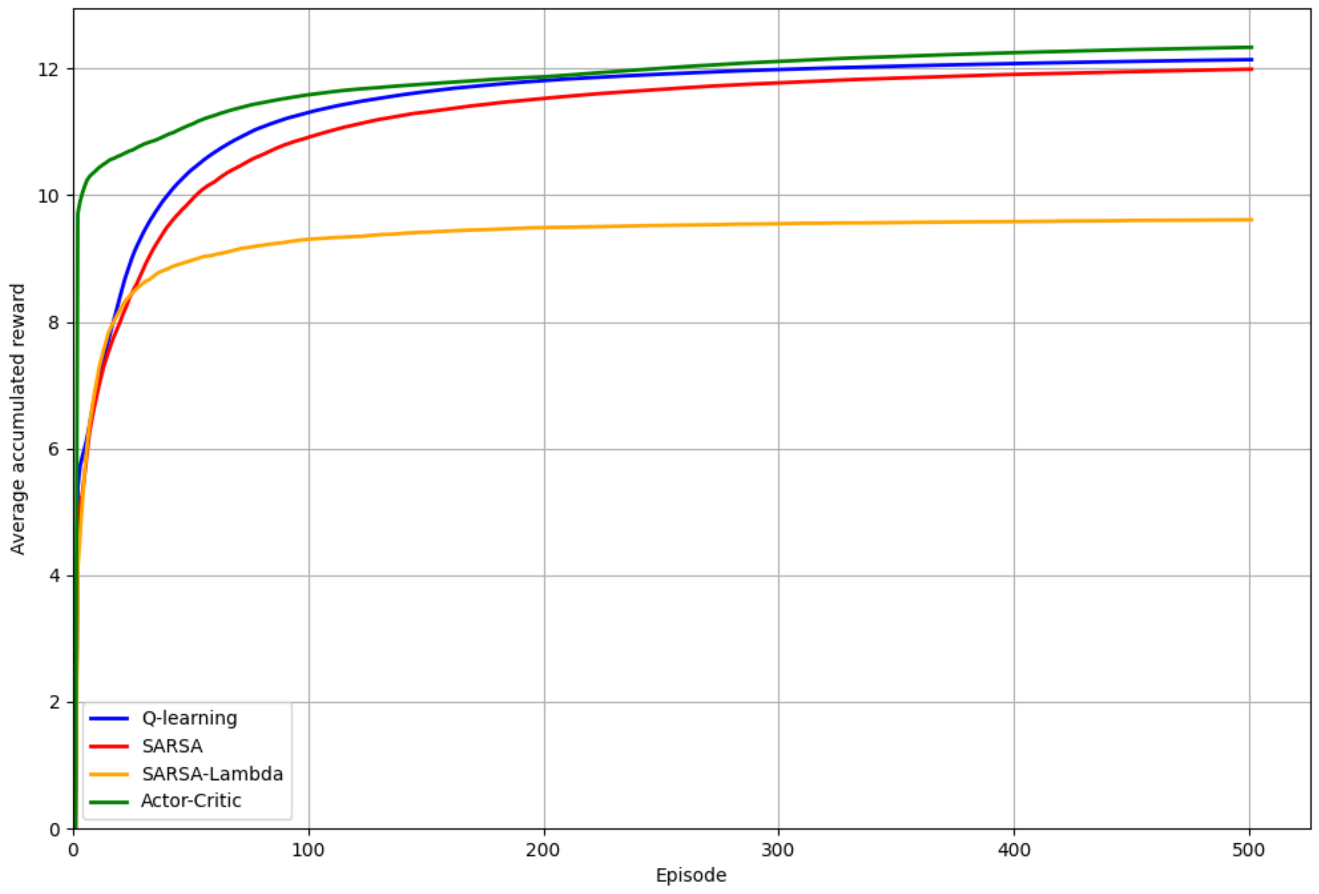

Figure 5: Average accumulated reward (in 10 independent runs) w.r.t episode number for all algorithms

Q-learning is an off-policy algorithm that estimates the value of the optimal policy by learning the action-value function.This estimation process is carried out regardless of the agent's current actions, which enables Q-learning to evaluate the potential of actions that have not been taken based on observed outcomes allowing the Q-learning algorithm to discover the optimal policy, even in environments high stochasticity environments.In the initial learning phase (Episode 0-100), the Q-learning curve displays a steep ascent, suggesting that the algorithm is rapidly identifying actions that lead to high rewards and making significant progress towards the optimal policy. This is indicative of Q-learning's preference of exploitation of gathered information in comparison to exploration to make substantial policy improvements without being constrained by the policy defined exploratory actions.During the mid-learning phase (Episode 100-300), the Q-learning curve's slope decreases,highlighting a reduction in the rate of improvement as the algorithm balances exploration (new actions to evaluate) and exploitation (using the known best actions) to stabilize converging policy.During the late learning phase (Episode 300-500), the Q-learning curve plateauing indicates that the Q-learning algorithm has converged to a stable policy that is close to the optimal policy for the stochastic environment and reward function.

From the graph, it can be observed that during the initial learning phase (Episodes 0-100), the SARSA curve exhibits a quick ascent. This highlights the on-policy nature of SARSA, which updates its policy based on the actions taken, including exploratory actions. The conservative rise of the SARSA curve suggests a cautious learning strategy of on-policy algorithms, which directly integrate actual exploratory actions into policy updates.In the mid-learning phase (Episodes 100-300), the SARSA algorithm continues to make progress with a slowing rate of improvement, indicative of the algorithm stabilizing and converging toward a policy that balances exploration with exploitation.The leveling off of the SARSA curve below the Q-learning curve suggests that the SARSA algorithms on-policy approach leads to a more conservative policy, possibly due to the reinforcement of sub optimal actions taken during exploration.The SARSA algorithm accumulates fewer rewards compared to a more exploitative strategy such as Q-learning, as the algorithm utilizes exploration for policy updates.During the late learning phase (Episode 300-500), the SARSA curve plateaus, indicating the algorithm has converged to a reliable stable policy, but not an optimal policy as the accumulated reward is not maximized.

From the graph, it can be observed that during the initial learning phase (Episodes 0-100),the SARSA((λ)) curve demonstrates a moderate ascent and then plateaus, indicating a relatively slower learning process due to the SARSA((λ)) algorithms on-policy nature which promotes balance of exploitation and exploration.During the mid-learning phase (Episode 100-300),the SARSA((λ)) curve indicates a relatively conservative update strategy due to the leveling off a lower accumulated reward suggesting a limitation or ineffectiveness of the eligibility traces in providing optimal action information from past experiences.In the late learning phase (Episodes 300-500),the SARSA((λ)) plateaus suggesting the algorithm's sensitivity to the stochasticity of the environment, leading to less optimal policy.

Actor-Critic algorithm utilizes the benefits of both on-policy and off-policy approaches by combining value function approximation (critic) with policy optimization (actor) to achieve a balance between the flexibility of policy-based method and the stability and efficiency of value-based method.During the initial learning phase (Episode 0-200), the Actor-Critic curve is similar to the Q-learning curve highlighting rapid increase in the accumulated reward. This indicates the actor's capability to quickly adapt policy based on the critic's accurate value assessment of policy's performance, enabling the actor to adjust the policy efficiently, effectively balancing exploitation and exploration of the environment.During the mid learning phase(Episode 100-300)the Actor-Critic algorithm continues to improve which indicates that the actor is refining policy based on new information provided by the critic adjusting policy in response to long-term rewards.In the late learning phase (Episodes 300-500), the Actor-Critic curve indicates convergence towards the optimal policy with highest average accumulated reward, suggesting that it has determined the most optimal policy for the stochastic environment. The actor-critic algorithm is able to perform well in a complex and stochastic environment as it can make incremental changes and refine policy based on the actor critic feedback loop.