Reinforcement Learning Algorithms – Final Experimental Report

Team Members

- **Youssef Ali Elsayed Ahmed** – 20100251
- **Bassant Awad Mohamed Mansour** – 22101405

---

1. Introduction

This report presents a comprehensive analysis and comparison of four reinforcement learning (RL) approaches—Dynamic Programming (DP), Q-Learning, SARSA, and REINFORCE—applied to the *JungleDash* grid-based environment. The objective is to evaluate each algorithm in terms of learning behavior, reward optimization, stability, efficiency, and robustness under identical experimental conditions.

A static version of the environment was used to ensure fairness and reproducibility. Performance metrics were collected over multiple episodes and analyzed both quantitatively and qualitatively.

---

2. Environment Summary

The JungleDash environment is an 8×8 grid world where an agent must reach a goal while collecting rewards and avoiding penalties.

Key characteristics: - State space: 64 discrete states - Action space: 4 actions (Up, Down, Left, Right) - Goal reward: +100 - Coin reward: +20 - Trap penalty: −20 (terminal) - Obstacle penalty: −1 (no movement) - Timeout penalty: −5

All experiments were conducted in **static mode**, ensuring a fixed layout across episodes and enabling a controlled comparison among algorithms.

---

3. Algorithms Evaluated

3.1 Dynamic Programming (Value Iteration)

Dynamic Programming is a model-based method that computes the optimal policy offline using complete knowledge of environment dynamics. It guarantees convergence to the optimal value function under deterministic conditions.

Strengths: - Guaranteed optimality - Stable and deterministic policy

Limitations: - Requires full environment model - Not suitable for unknown or large-scale environments

---

3.2 Q-Learning

Q-Learning is a model-free, off-policy temporal-difference algorithm. It learns optimal action values by interacting with the environment and updating estimates using the Bellman optimality equation.

Strengths: - Learns optimal policy independently of behavior policy - Effective in unknown environments

Limitations: - High variance during learning - Can be unstable without sufficient exploration control

---

3.3 SARSA

SARSA is a model-free, on-policy temporal-difference method. Unlike Q-Learning, it updates action values using the action actually taken under the current policy.

Strengths: - More conservative and stable learning - Safer in stochastic or risky environments

Limitations: - Slower convergence - Policy-dependent learning may limit optimality

---

3.4 REINFORCE

REINFORCE is a policy-gradient, Monte Carlo algorithm that directly optimizes the policy by maximizing expected return.

Strengths: - Direct policy optimization - Naturally handles stochastic policies

Limitations: - High variance in updates - Requires many episodes for convergence

---

4. Experimental Results Analysis

4.1 Reward Comparison

- **Dynamic Programming** achieved the highest average and final rewards, demonstrating near-optimal behavior.
- **Q-Learning** showed significant improvement over time but suffered from high variance.
- **SARSA** achieved moderate rewards with more stable learning than Q-Learning.
- **REINFORCE** produced the lowest rewards due to high variance and delayed updates.

Ranking by final average reward: 1. Dynamic Programming 2. Q-Learning 3. SARSA 4. REINFORCE

4.2 Convergence and Stability

- DP converged immediately since the policy is computed offline.
- Q-Learning exhibited oscillations due to off-policy updates.
- SARSA showed smoother but slower convergence.
- REINFORCE required many episodes and remained unstable.

4.3 Efficiency (Steps and Penalties)

- DP and SARSA showed similar step counts but DP accumulated more penalties due to deterministic behavior.
- Q-Learning incurred fewer penalties per episode.
- REINFORCE had the highest penalties, reflecting inefficient exploration.

4.4 Success Rate

- Only REINFORCE achieved a non-zero success rate (≈6%), indicating occasional successful trajectories.
- Other algorithms prioritized reward accumulation over direct goal completion.

---

5. Comparative Summary

Algorithm	Model-Based	Policy Type	Convergence	Stability	Performance
DP	Yes	Deterministic	Immediate	Very High	Excellent
Q-Learning	No	Off-policy	Moderate	Medium	Good
SARSA	No	On-policy	Slow	High	Fair
REINFORCE	No	Policy Gradient	Very Slow	Low	Weak

---

6. Discussion

The results clearly show that **Dynamic Programming** outperforms all other methods in static, fully known environments. However, its reliance on a complete environment model limits its real-world applicability.

Model-free methods (Q-Learning and SARSA) demonstrate strong adaptability but require careful tuning to balance exploration and exploitation. REINFORCE, while theoretically powerful, suffers from high variance and inefficient learning in small discrete environments.

---

7. Conclusion

This experiment highlights the trade-offs between model-based and model-free reinforcement learning approaches: - **DP** is ideal when environment dynamics are known and fixed. - **Q-Learning** is effective for learning optimal behavior in unknown environments but may be unstable. - **SARSA** offers safer and more stable learning at the cost of optimality. - **REINFORCE** is better suited for complex or continuous domains rather than small grid worlds.

Overall, **Dynamic Programming** was the best-performing algorithm in this experimental setup, while **Q-Learning** emerged as the most practical model-free alternative.

---

8. Final Remark

The study demonstrates how algorithm choice should be guided by environment characteristics, available information, and performance requirements rather than reward maximization alone.