Markov Decision Processes: An Application to RISK

Daniel Yao

Johns Hopkins University

24 April 2025

Introduction

Markov Decision Process [3]

- A *Markov Decision Process* (MDP) is a tuple (S, A, P, R, γ).
- S is the *state space*.
- A is the *action space* where $A(s)$ is the action space of state s.
- $P : S \times A \times S \to [0, 1]$ is the *transition function* where $P(s' \mid s, a)$ is the probability of transitioning to state s' given that action a is taken in state s.
- $R : S \times A \times S \to \mathbb{R}$ is the *reward function* where $R(s, a, s')$ is the reward received after transitioning to state s' from state s by taking action a.
- $\gamma \in [0, 1]$ is the *discount factor*.

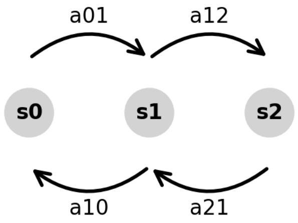
Introduction

Markov Property [3]

- An MDP has the *Markov property* that

$$P(s_{t+1} \mid s_t, a_t, s_{t-1}, a_{t-1}, \ldots) = P(s_{t+1} \mid s_t, a_t)$$

Transition Diagram [3]



Introduction

Policy [3]

- A *policy* π is a function $\pi : A \times S \to [0, 1]$ where $\pi(a \mid s)$ is the probability of taking action a in state s.
- A *deterministic policy* π is a function $\pi : S \to A$ where $\pi(s)$ is the action taken in state s.

Discounted Return [3]

- The *discounted return* G_t at time t is the sum of all future (discounted) rewards. That is,

$$G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}.$$

Introduction

State-Action Value Function [3]

- The *state-action value function* $Q_\pi : S \times A \to \mathbb{R}$ is the expected discounted return of taking action a in state s and following policy π thereafter. That is,

$$Q_\pi(s, a) = \mathbb{E}_\pi \left[G_t \mid s_t = s, a_t = a \right].$$

Optimal Policy [3]

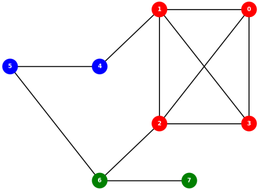
- An *optimal policy* π^* is a policy that maximizes the state-action value function such that

$$\pi^* = \arg\max_\pi Q_\pi(s, a)$$

Methods

RISK (Game) [2]

- We described a simplified 2-player version of RISK.
- There are 3 continents $\{0, 1, 2, 3\}$, $\{4, 5\}$, and $\{6, 7\}$.
- Each territory must always be occupied by at least 1 army.
- A turn has 4 phases: reinforce, attack, fortify, and end turn.



Methods

Reinforce [2]

- If player 1 controls t territories and has continent bonus c, then he receives $\min(\lfloor t/3 \rfloor + c, 1)$ armies.
- He may distribute these armies anywhere in his territories.

Attack [2]

- Player 1 commits k armies to attack an adjacent territory with m armies controlled by player 2.
- Player 1 rolls $\min(k, 3)$ dice and player 2 rolls $\min(m, 2)$ dice.
- The highest die of each player is compared and the second highest die of each player is compared. Player 2 wins ties.
- For each die lost, the player loses one army.
- If victorious, player 1 may attack again with his remaining troops.

Methods

Fortify [2]

- Player 1 may redistribute his armies.
- He may only move armies through his contiguous territories.

End Turn

- Player 1 passes the turn to player 2.

Methods

Environment [4]

- The environment was implemented with Gymnasium.
- The *state space* S was

$$S = (\{1,2\} \times \{1,\ldots,40\})^8$$

  where $s = (x_1, y_1, \ldots, x_8, y_8)$ means that player x_i controls y_i armies in territory i.
- The *action space* A was the set of all legal moves in 4 phases: reinforce, attack, fortify, and end turn.
- The *transition function* P was defined by the combat mechanics.

Methods

Environment

- The *reward function* R was based on the difference between the reinforcement values of the two players.
- Intermediate rewards were scaled by a factor of 0.1.
- The *discount factor* was $\gamma = 0.99$.

Methods

Reinforcement Learning [1] [3] [6]

- Double deep Q-Learning was implemented with Keras.
- An epsilon-greedy policy was used.
- The agent learned to play against a random opponent.
- 256 games of 20 turns were played.
- Starting positions were randomly generated.

Test

- The agent played 32 games of 20 turns against a random opponent.
- The victor and final reward were recorded.

Results

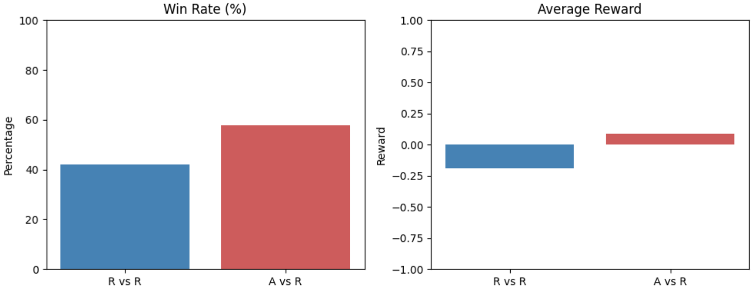
Random versus Random

- 42.19% win rate
- -0.1875 average reward.

Agent versus Random

- 57.81% win rate
- $+0.0893$ average reward.

Results

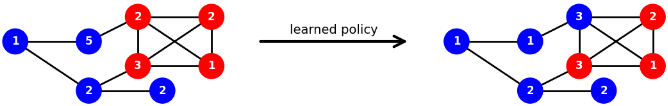


Win Rate (%)
Percentage
100
80
60
40
20
0
R vs R
A vs R
Average Reward
Reward
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
R vs R
A vs R

Discussion

Sun Tzu said... [5]

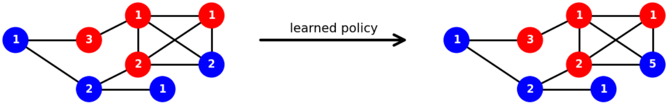
- "So in war, the way is to avoid what is strong, and strike at what is weak."
- "Do not swallow a bait offered by the enemy."



Discussion

Sun Tzu said... [5]

- "Move only if there is a real advantage to be gained."
- "Whether to concentrate or to divide your troops, must be decided by circumstances."



Conclusion

Conclusion

- Our MDP + Double DQN method yielded modest improvements over a random agent.
- Still, the agent would lose against a heuristic agent or a human player.

Sun Tzu said... [5]

- "To know your Enemy, you must become your Enemy."
- Train by self-play.

References

[1] Chollet, F. et al. (2015). Keras. `https://keras.io`.

[2] Parker Brothers (1993). *Risk: The Classic World Domination Game*. Tonka Corporation.

[3] Puterman, M. L. (2014). *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons.

[4] Towers, M., Kwiatkowski, A., Terry, J., Balis, J. U., De Cola, G., Deleu, T., Goulão, M., Kallinteris, A., Krimmel, M., KG, A., et al. (2024). Gymnasium: A standard interface for reinforcement learning environments. *arXiv preprint arXiv:2407.17032*.

[5] Tzu, S. (1998). *The Art of War*. Project Gutenberg.

[6] Van Hasselt, H., Guez, A., and Silver, D. (2016). Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30(1).

Appendix

Code Availability

- github.com/dyao13/risk_agent