cs339: Artificial Intelligence

Project 4: Reinforcement Learning

1 Project Overview

The goal of this project is to *solve a game*. By solving a game, we mean to determine

1. the optimal strategy and
2. the ”score” you can expect to earn when playing optimally.

We will apply various algorithms related to Reinforcement Learning to tackle the same problem and compare the relative performance. You will be working in a group of 3 or 4 people, each of whom will solve the problem with a different algorithm. Your team will submit one research paper that compares the performance of each algorithm.

The learning goals of this project are

- To understand problems that seek optimal behavior (the basic Reinforcement Learning framework).
- To learn more deeply and to apply Dynamic Programming to a challenging problem.
- To learn and apply Monte Carlo methods to a challenging problem.
- To learn and apply the SARSA and Q-Learning reinforcement learning algorithms to a challenging problem.
- To develop group and project management techniques to achieve a successful project by the deadline.
- To gain experience writing group research papers.

2 Problem Domain

Your group will all tackle the problem of learning to play Chutes and Ladders optimally (the game is also known as Snakes and Ladders in some other countries). We will modify the game since the original version has no strategic decisions. The goal is to train an agent to play the modified game optimally.

Chutes And Ladders is a childhood game in which the goal is to move your token from Square 0 (off the bottom, left corner of the board) to Square 100. See Figure 1 for an image of the game board. The first player to move their token to Square 100 wins the game. Each move consists of spinning a ”spinner” which has the numbers 1 through 6 written on it; the player moves their token ahead the designated number of moves. If the player lands exactly on the bottom of a ladder square (see Square 4 for example), they move immediately to the top of the ladder. Conversely, if the player lands exactly on the top of a chute square (see Square 16 for example), the must move their token to the bottom of the chute. A player does not need an exact spin to land on Square 100; any spin that reaches that square is adequate for the final move (for example, a player may spin a 6 from Square 97 and win the game).

We will change the game slightly using dice instead of a spinner. As the game is formulated, it is all luck based on the spin. Our game will introduce an element of strategic choice. Consider that each player has four dice (specified below)[1]. On each move, the player may select any one of the dice, roll that dice, and use the result for their move. Because the dice have different numbers, players may strategically choose a dice that has a higher probability of avoiding a chute or reaching a ladder.

Dice	**Faces**					
black	4	4	4	4	0	0
red	6	6	2	2	2	2
green	5	5	5	1	1	1
blue	3	3	3	3	3	3

The strategy is to learn which dice to play at each square on the board so as to minimize the future number of steps. Because the game is stochastic, the ”future number of steps” is measured as an expected value, an average future number of steps. Notice that strategies and also expected steps-to-goal for one square are dependent on other squares. There is also something special about this game which makes it more challenging to solve, especially with Dynamic Programming.

3 Group Management

The goal of this project is to apply a number of different techniques to ”solve” the game. By solving the game, we mean determining the optimal average number of

[1]These are called Efron Dice and have special nontransitive properties – search Wikipedia for nontransitive dice.

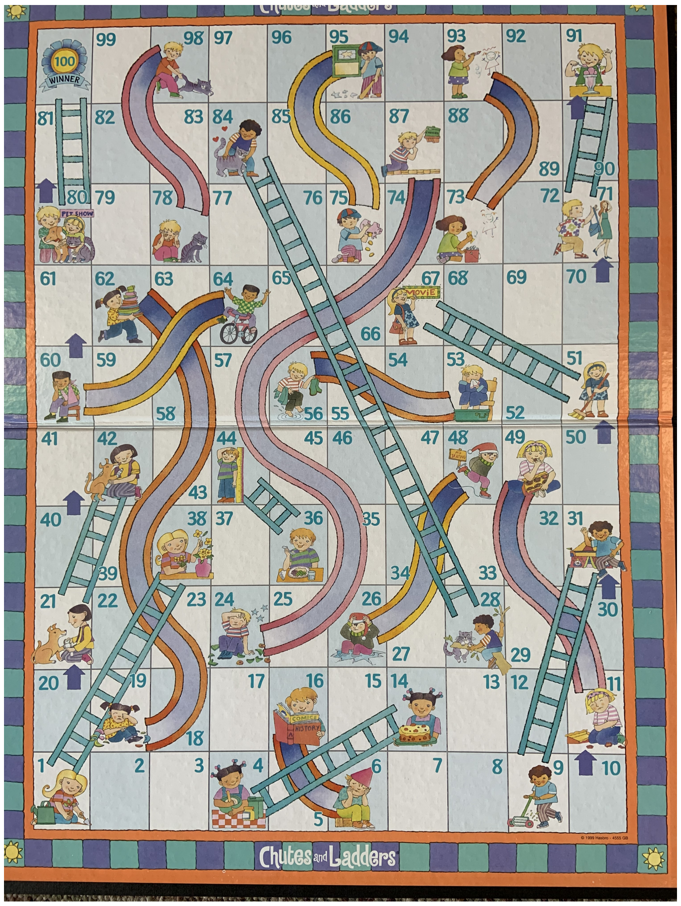

Figure 1: Chutes and Ladders Game Board

moves expected to complete the game and the optimal strategy for selecting which dice on each possible square of the game. Each team member in your group should solve this game using a different technique. Here are the techniques:

1. Dynamic Programming:
   In this method, we use Dynamic Programming or other similar tools to solve the game optimally off-line.

2. Monte Carlo Methods:
   We use random simulations to determine average moves for each possible state/action combination.

3. Reinforcement Learning:
   We use RL techniques to learn online how to play optimally. I suggest implementing both on policy (SARSA) and off policy (Q-Learning) to determine if there is a difference.

If your group has four members, each person will take one of the above algorithms (one person for SARSA, one person for Q-Learning). If your group has three members, the last person will complete both Q-Learning and SARSA.

A solution to the game consists of the Q-values (state/action values), the V values (state values) and Policy (strategy). Each of the above methods may compute the same solution (Q,V,P) or have slightly different outcomes.

You will want to measure the following properties of your algorithms:

- How ”quickly” do the methods arrive at a best solution? How quickly do they converge on a good solution?
- Do they find the optimal solution? Do they find the optimal solution repeatedly?
- How close to the optimal solution do they arrive?
- How will you measure learning speed? How will you measure algorithm performance (optimality)?
- How will you display the policy learned? How will you display the learning performance?

You will want to assess your algorithm’s performance. There are two interesting assessment metrics. First is ”learning time” – how many trials of experience does it take to form a good/optimal policy? This metric isn’t applicable to DP methods since they are off-line. You might also look at computing time/cycles for learning; then you can fold in the computing-intense DP methods. Second you want to assess the optimality of the final policy. How well does it solve the problem? Compare performance metrics across the different algorithms.

The project *must* be a group writing effort as well. You will want to coordinate and divide the writing work among the team members. Do not delegate the writing to one team member. An explicit goal of this project is to learn to co-write a paper with a small team.

4 Deliverables and Deadlines

Item	Deadline	Description
Algorithm	—	Submit a working notebook for each of the four algorithms.
Visuals	—	Complete the graphs and charts for comparing performance of each algorithm.
Paper	—	Submit a final latex paper for the whole group.

The final deliverable will be a single paper with a thorough analysis of the whole project, including results. In an appendix, you should include a list of tasks that each person contributed towards. This will likely be a longer paper than the previous ones.

5 Group Assignments

- **Group 1:** Junye, Yubo, Derek
- **Group 2:** Giogi, Lucas, Jin
- **Group 3:** Henri, Liam, Juntao, Brandon
- **Group 4:** Khoi, Mark, Philipp, Amna
- **Group 5:** Anub, Yash, Jeff, Jash

Please all register under notebowl in your groups.