

Homework 4: Pacman versus the Ghosts

Due September 28th at 10pm

Part 1: Adversarial cheese search

Concentration is a game where the players take turns listing items in a particular category. If you cannot think of an item in the category within a time limit, you lose. You also lose if you say an invalid item or repeat an item that has previously been named.

There are many variants of this game. We will focus on a variant where each player's item must start with the same letter as the last item ended with.

Task 1: Saying cheese

Assume that the category is cheese. You and your fellow player know only 5 kinds of cheese: edam, Emmental, Munster, paneer, and raclette. **Draw out the game tree for this small vocabulary list.** You should mark wins and losses on terminal nodes.

Assume that you start: you have the choice between all five cheeses for your first move. I recommend labeling the nodes with players and the transitions between them with words.

Task 2: Minimaxing cheese

So far, we have been assuming that our opponent knows every possible word in the category. MiniMax is the optimal adversarial strategy assuming the other player has the same information as you and acts optimally.

But some of the cheeses listed above might be hard to come up with under pressure. Could you **improve** on the basic MiniMax strategy (i.e., make it more efficient) by guessing how your opponent might not be an optimal player? if you modeled an imperfect player? If so, **how would you change the algorithm?**

Part 2: Pacman and enemies

We return now to our Pacman game to implement adversarial search strategies. Unlike last week, you now have adversaries: ghosts that will eat Pacman if he makes contact with them.

Refer to the UC Berkeley assignment for instructions on how to run the Pacman game simulation: UC Berkeley Project 2.

Task 3: Minimax

Implement minimax for Pacman in multiAgents.py. This involves generalizing our minimax algorithm to games with two players: there are now multiple min layers for each max, since there are multiple ghosts.

You'll probably need to write some helper functions. I structured my approach as follows:

`getAction`: calls `self.minimax` and returns the result

`minimax`: handles the logic for switching between layers in the tree. Takes `self`, `gameState`, `player`, and `depth` as arguments. Figures out if the game is going to continue and calls `getValue` to find the best value to take. Returns a pair consisting of the best value and its associated action.

`getValue`: handles a single layer in the tree. Takes `self`, `gameState`, `player`, and `depth` as arguments. For the given state, considers all actions and recursively calls `minimax` to calculate their values. Returns a pair consisting of the best value and its associated action.

You can follow this approach, or you could follow the pseudo-code given in class, which splits the minimax function into separate min and max functions.

Part 3: AI Needs You reflections

Task 4: Reading Reflection Question 1

Describe and discuss how President Eisenhower reacted to the trade-offs involved in funding the space race.

Describe and discuss how President Kennedy reacted to the trade-offs involved in funding the space race.

Task 5: Reading Reflection Question 2

President Eisenhower delivered the following remarks, which are quoted in this chapter (p. 41):

Every gun that is made, every warship launched, every rocket fired signifies, in the final sense, a theft from those who hunger and are not fed, those who are cold and are not clothed. The world in arms is not spending money alone. It is spending the sweat of its laborers, the genius of its scientists, the hopes of its children.'

Reflect on the remark above and how it might relate to AI development today. **How might an AI optimist respond to Eisenhower's remark? How might an AI pessimist respond?**