CS188 Fall 2015 Section 2: Non-Classical Search and Games

1 Battleship: Partial Observability

You are playing a modified version of Battleship on a 3x3 board. Your opponent places a length-3 ship somewhere on the board, unknown to you. Ships can be oriented vertically or horizontally.

Every turn, you choose one location on the board to attack. If you attack a location containing part of the ship, you receive the percept "Hit" and win the game.

If you miss, your percept will be "Near" if the ship is one unit away from your attack location, and "Far" if it's two units away. Your goal is to use *offline belief state search* to come up with a plan to hit your opponent's ship with the fewest attacks.

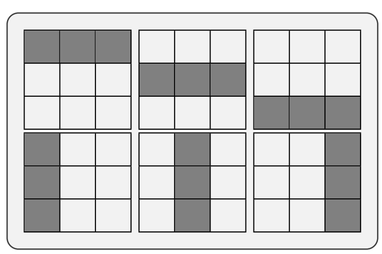
1. How would the *physical state* be represented in this game? What is the size of the state space? The relevant information the state needs to store is

   - The location of your opponent's ship. The ship can be vertically in any column or horizontally in row, which is 6 possible orientations.
   - The most recent attack attempt. With the location of the ship, this allows the game to generate our percept ("Near", "Far", or "Hit"). This can be in one of 9 locations, or there could have been no attack made, meaning there are 10 possibilities.

   Putting that together, we get a state space of size $6 \times 10 = 60$ states.

2. What is the initial belief state (before you make any actions)? How many elements does it have?

   The initial belief state is the set of all physical states you haven't ruled out as impossible. Since you know you haven't made any attacks yet, there should be no attacks recorded in any of the physical states in your belief state. The states should also include every possible orientation of the ship, since you don't know where it is. This gives a size 6 belief state that looks like this:

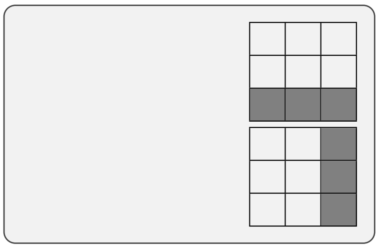


3. What is the size of the belief state space?

   There are 60 possible physical states, and the belief state space is the powerset of the physical states, so there are 2^{60} possible belief states.

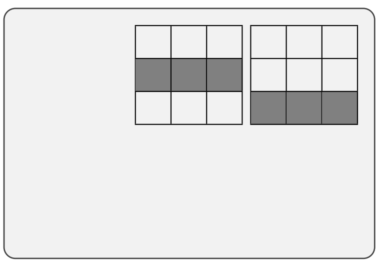
4. A *reachable* belief state is any belief state that can result from a sequence of actions starting at the initial belief state. Below, we show two possible belief states with no attack information marked. Your job is to determine the sequence of attack attempts and corresponding percepts that make each belief state reachable, or else explain why the belief state is not reachable.

   (a)

   This belief state can be reached by attacking the top left square and receiving a percept of "Far". No other physical states would give that percept for that attack.

   (b)

   This belief state is unreachable. Let's call the physical state on the left A and the state on the right B.

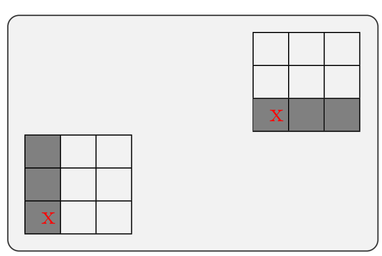
   If we attacked any square in the top row, we would get a percept of "Near" if the physical state were A, and "Far" if the physical state were B. This would force us to eliminate one or the other from the resulting belief state.

   Attacking any square in the bottom row, A would give a percept of "Near" while B would give a percept of "Hit". Similarly, attacking a square in the middle row would give "Hit" for A and "Near" for B. Both of these options would also force us to eliminate one of A or B.

5. From the initial belief state, suppose you choose to attack the bottom left corner. What percepts could you experience at this point? What belief states would they lead to?

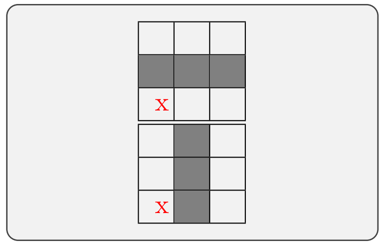
   There are three possible percepts:

   - "Hit": If we experience this percept, our new belief state would look like this:

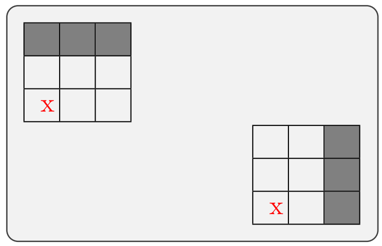
     

     All the states in this belief state are goal states, so the belief state is also a goal state.

   - "Close": If we experience this percept, our new belief state would be



- “Far”: If we experience this percept, our belief state would be



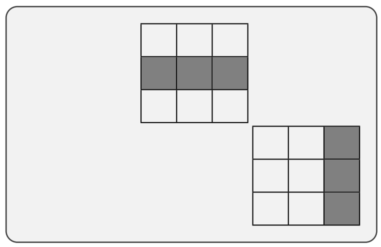
6. What is the optimal contingent plan if we begin by attacking the bottom left square?

If the percept is “Hit”, we’ve already won. If it’s “Close”, we should attack the middle square to guarantee a win. If it’s “Far”, we should attack the top right square to guarantee a win.

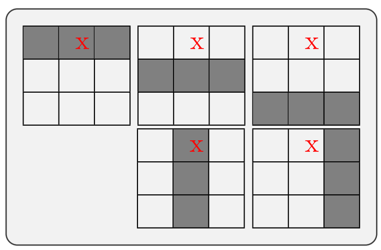
2 More Battleship: Nondeterminism

Now, we allow your opponent to move their ship if your attack misses. If the ship is oriented horizontally, it may move one unit up or down. If it's oriented vertically, it may move one unit right or left. Your new percepts are "Hit", "Far", "Move" if the ship was one away when you attacked and has moved, and "Stay" if the ship was one away when you attacked and did not move.

1. Suppose you are currently in this belief state (attack sequence not indicated):



What is your belief state immediately after you attack the top middle square, before seeing any percepts? Because the ship can move, it could be in any location except the first column:



2. Was your resulting belief state larger or smaller than your initial belief state? Why? The resulting belief state was larger than the initial belief state, because each physical state could result in more than one possible successor state due to your action. In a deterministic environment like our previous example, this would never happen. Each physical state would result in exactly one successor state, meaning the successor belief state would always be smaller than or equal to the initial belief state.

3. What is the difference between partial observability and non-determinism?

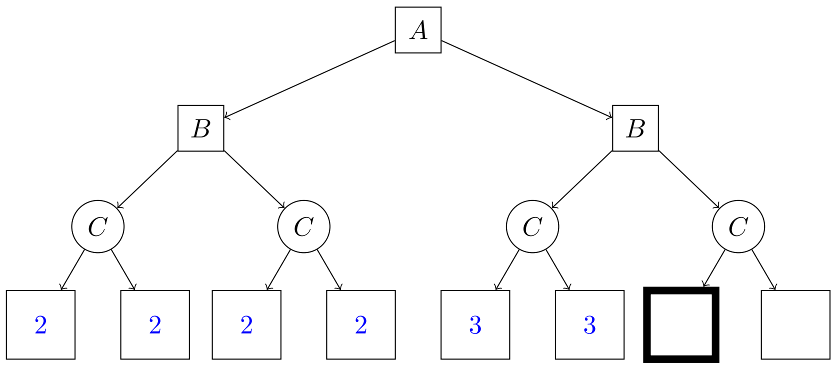
   Partial observability refers to *fixed aspects of the environment* that are unknown to the agent. Partial observability is what necessitates belief state search, because you have to consider all the possible ways the world could be when making your plan.

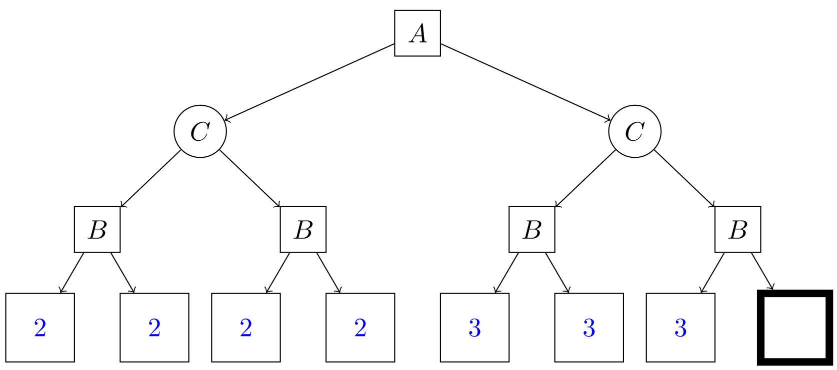
   Non-determinism refers to *the agent's actions* having more than one possible consequence. Non-determinism is what necessitates contingent search, because you have to have a plan for every way your actions could turn out.

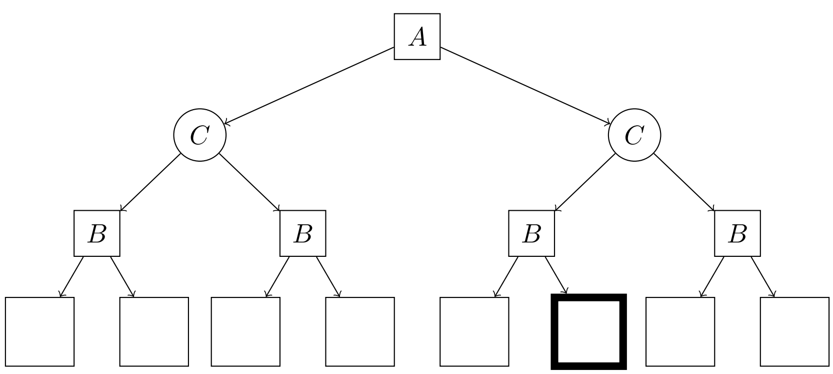
3 Alpha Beta Expinimax

In this question, player A is a minimizer, player B is a maximizer, and C represents a chance node. All children of a chance node are equally likely. Consider a game tree with players A, B, and C. In lecture, we considered how to prune a minimax game tree - in this question, you will consider how to prune an expinimax game tree (like a minimax game tree but with chance nodes). Assume that the children of a node are visited left to right.

For each of the following game trees, give an assignment of terminal values to the leaf nodes such that the bolded node can be pruned (it doesn't matter if you prune more nodes), or write "not possible" if no such assignment exists. You may give an assignment where an ancestor of the bolded node is pruned (since then the bolded node will never be visited). You should not prune on equality, and your terminal values *must* be finite (including negative values).







Not possible. At the bolded node, the minimizer can guarantee a score of at most the value of its left subtree. However, since we have not visited all the children of C, there is no bound on the value that C could attain. So, we need to continue exploring nodes until we can put a bound on C's value, which means that we have to explore the bolded node.