CS 520: α-β Pruning with Chance

16:198:520

In class, we discussed a search algorithm for adversarial settings that tried to limit the amount of search space that needed exploring by bounding the relative utilities available down certain branches. If the **MAX** player has a utility of 1 down one branch, and a utility of *no better than* 0 down an alternate branch, the **MAX** player does not need to search the complete search tree below this second option to know that the first branch is a better bet.

α-β pruning in this way can potentially cut down a lot of effort in evaluating the search tree and determining moves - but consider the problem of *expectiminimax* search trees, where there is an additional 'chance' element, and evaluating the utility of a given node may depend on evaluating the utilities along all the chance or random branches. Can we adapt α-β pruning in this context to try to cut down on the overall work we have to do?

Consider a simple game where the **MAX** player has a choice of two moves $\{a_1, a_2\}$, the **CHANCE** player then takes one of two possible branches $\{c_1, c_2\}$ (whose **known** probabilities may depend on the move **MAX** took), the **MIN** player then has a choice of two moves $\{b_1, b_2\}$. At this point, either **MAX** wins the game, **MIN** wins the game, or there is a tie (representing utilities $+1, -1, 0$ respectively). An important note in advance: while the specific utility of the terminal state may change from terminal state to terminal state, we know in advance before the search tree is assessed that the final utilities are bound between -1 and $+1$. (The picture you should draw for this game tree has one initial state, two nodes below it, four possible nodes below that, and then eight possible nodes below that.)

Consider walking through the minimax search process from the perspective of player **MAX**:

- **MAX** estimates the expected utility of a_1, a_2 to each be in the interval $[-1, 1]$ - this is based on the initial knowledge that all final utilities are bounded.
- **MAX** explores the possible game tree based on taking action a_1.
- In typical depth first search fashion, **MAX** probes down c_1, then down the action b_1 to reach the terminal state (a_1, c_1, b_1).
- At this point, **MAX** assesses the utility of state (a_1, c_1) to be in the range

$$U(a_1, c_1) \in [-1, U(a_1, c_1, b_1)]. \tag{1}$$

  This captures the fact that player **MIN** may find a 'better' option with one of the remaining two actions, but surely the 'worst' utility available at state (a_1, c_1) is bound from below by -1 and above by $U(a_1, c_1, b_1)$, whatever the result of this terminal state was.

- Player **MAX** additionally knows that

$$U(a_1) = \mathbf{P}(c_1|a_1)U(a_1, c_1) + \mathbf{P}(c_2|a_1)U(a_1, c_2). \tag{2}$$

  Plugging in the bounds we have now assessed for $U(a_1, c_1)$ and the prior knowledge bounds on the other utility, we have

$$U(a_1) \in [\mathbf{P}(c_1|a_1)(-1) + \mathbf{P}(c_2|a_1)(-1), \mathbf{P}(c_1|a_1)U(a_1, c_1, b_1) + \mathbf{P}(c_2|a_1)(1)], \tag{3}$$

  or, observing that $\mathbf{P}(c_1|a_1) + \mathbf{P}(c_2|a_1) = 1$,

$$U(a_1) \in [-1, 1 - \mathbf{P}(c_1|a_1)(1 - U(a_1, c_1, b_1))], \tag{4}$$

- At this point, we have percolated the gained information up as high as we can / as is necessary. We can continue searching the tree, in particular, looking at the remaining **MIN** action from (a_1, c_1) gives us

$$U(a_1, c_1) = \mathbf{min}\left[U(a_1, c_1, b_1), U(a_1, c_1, b_2)\right]. \tag{5}$$

- This percolates upwards:

$$U(a_1) \in [\mathbf{P}(c_1|a_1)U(a_1,c_1) + \mathbf{P}(c_2|a_1)(-1), \mathbf{P}(c_1|a_1)U(a_1,c_1) + \mathbf{P}(c_2|a_1)(1)], \tag{6}$$

  or noting again that $\mathbf{P}(c_2|a_1) = 1 - \mathbf{P}(c_1|a_1)$,

$$U(a_1) \in [-1 + \mathbf{P}(c_1|a_1)(1 + U(a_1,c_1)), 1 - \mathbf{P}(c_1|a_1)(1 - U(a_1,c_1))], \tag{7}$$

- To complete the assessment of the value of action a_1, it remains to explore the branch of the game tree down the chance action c_2. Player **MAX** arrives at the conclusion:

$$U(a_1,c_2) = \mathbf{min}\left[U(a_1,c_2,b_1), U(a_1,c_2,b_2)\right], \tag{8}$$

  which gives us the final result that

$$U(a_1) = \mathbf{P}(c_1|a_1)\mathbf{min}\left[U(a_1,c_1,b_1), U(a_1,c_1,b_2)\right] + \mathbf{P}(c_2|a_1)\mathbf{min}\left[U(a_1,c_2,b_1), U(a_1,c_2,b_2)\right]. \tag{9}$$

This tracks roughly how the game search tree is explored in the establishing of the value for node a_1. If the search continues down action a_2, searching the first chance branch to completion, we will then have that

$$U(a_2,c_1) = \mathbf{min}\left[U(a_2,c_1,b_1), U(a_2,c_1,b_2)\right], \tag{10}$$

and percolating this result upwards,

$$U(a_2) \in [-1 + \mathbf{P}(c_1|a_2)(1 + U(a_2,c_1)), 1 - \mathbf{P}(c_1|a_2)(1 - U(a_2,c_1))], \tag{11}$$

At this point, the branch along chance action c_2 remains unexplored off node a_2. But does it need to be explored? If we had that the upper bound on $U(a_2)$ was less than the calculated value of $U(a_1)$, the **MAX** player is never going to take action a_2 over a_1. Similarly, if the lower bound on $U(a_2)$ is greater than the calculated value of $U(a_1)$, the **MAX** player is never going to take action a_1 over a_2. This gives us two potential 'early termination conditions':

$$\textbf{if } 1 - \mathbf{P}(c_1|a_2)(1 - U(a_2,c_1)) \leq U(a_1) \textbf{ terminate and take } a_1, \tag{12}$$

and

$$\textbf{if } -1 + \mathbf{P}(c_1|a_2)(1 + U(a_2,c_1)) \geq U(a_1) \textbf{ terminate and take } a_2. \tag{13}$$

If either case is met, this expected version of α-β pruning will cut down on the over all amount of work that the player needs to do in searching. Rearranging, for a simple game of this structure we can terminate the search early if we meet the condition:

$$1 - \frac{1 - U(a_1)}{\mathbf{P}(c_1|a_2)} \geq U(a_2,c_1) \textbf{ OR } U(a_2,c_2) \geq \frac{1 + U(a_1)}{\mathbf{P}(c_1|a_2)} - 1 \tag{14}$$

1 An Actual Example Game

Suppose that $\mathbf{P}(c_1|a_1) = 1/2, \mathbf{P}(c_2|a_1) = 1/2$, and that $\mathbf{P}(c_1|a_2) = 4/5, \mathbf{P}(c_2|a_2) = 1/5$. Additionally, take

$$\mathbf{min}\left[U(a_1,c_1,b_1), U(a_1,c_1,b_2)\right] = -1 \tag{15}$$

$$\mathbf{min}\left[U(a_1,c_2,b_1), U(a_1,c_2,b_2)\right] = 0. \tag{16}$$

In this case, We then have

$$U(a_1) = \frac{1}{2}(-1) + \frac{1}{2}(0) = -0.5. \tag{17}$$

Plugging this into the early termination condition, we have

$$-0.875 \geq U(a_2, c_1) \textbf{ OR } U(a_2, c_1) \geq -0.375. \tag{18}$$

That is, if we explore branch a_2, c_1 and discover that the utility is particularly small, or particularly large, we don't actually have to worry about exploring branch a_2, c_2 - we can logically determine the optimal move to take at this point.