The Hebrew University of Jerusalem Introduction to Artificial Intelligence Problem Set 2 Adversarial search, Knowledge representation

Due Date 28.05.2020

0.1 Run minimax on a tree

[25 points]

Consider the two-player game tree shown below. This tree is a binary tree and its leaf nodes are all at depth 4. The two players, MAX and MIN, play in turn. At each turn the playing player must choose between moving "left" or "right". MAX is the first player and uses an evaluation function e that maps each state of the game (node of the tree) to a positive integer. A larger e at a node, the more promising this node is for MAX. The value of e at each leaf is shown inside the leaf node. For the purpose of this problem, we give a label to each intermediate node: the root is labeled by A, its two children by B and C, etc. The leaves at depth 4 have not been assigned any labels.

1. In figure 1, fill each intermediate node with the value that is backed-up by the Minimax algorithm. According to the backed-up values, what move MAX should play: left or right?

0.2 Run alpha-beta pruning on a tree

1. In figure 2 fill each intermediate node with the alpha and beta values computed by the Alpha-Beta pruning algorithm. In addition, clearly mark all the nodes that are not examined by this algorithm by filling them in black (or any color available to you) or with a clearly visible X. Non-examined nodes may include leaf nodes and intermediate nodes.

2. According to the alpha and beta values what move MAX should play: left or right? In general, is the best move computed by the Alpha-Beta pruning algorithm guaranteed to always be the same as the one computed by the Minimax algorithm?

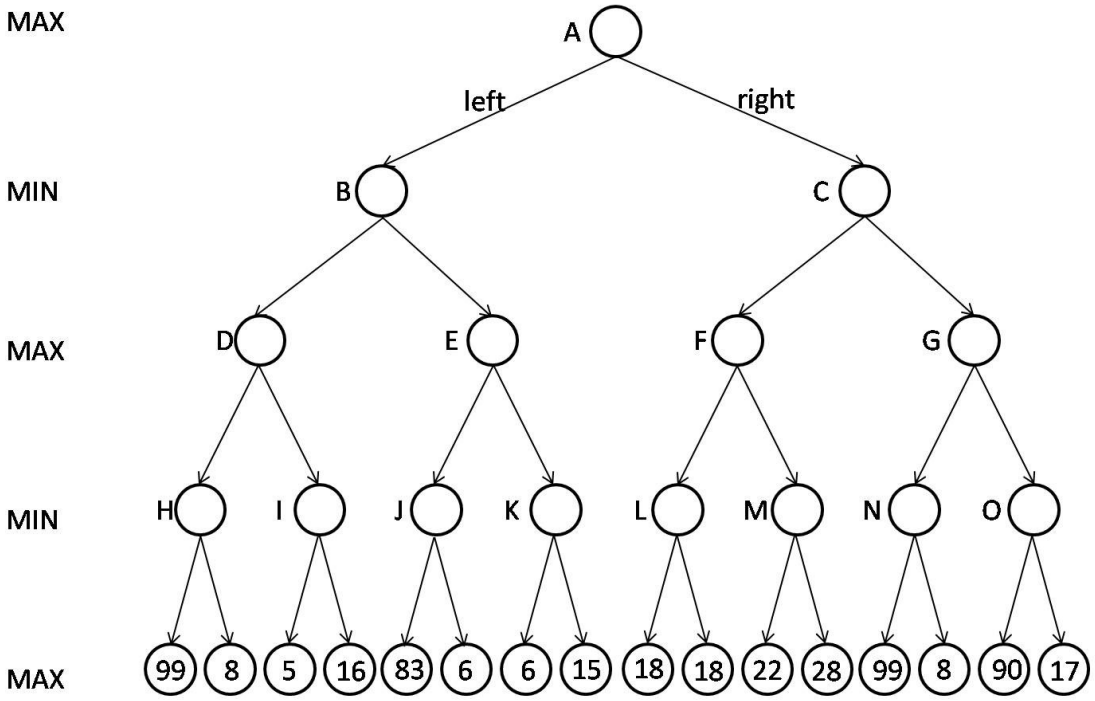

Figure 1: Fill in the minimax values.

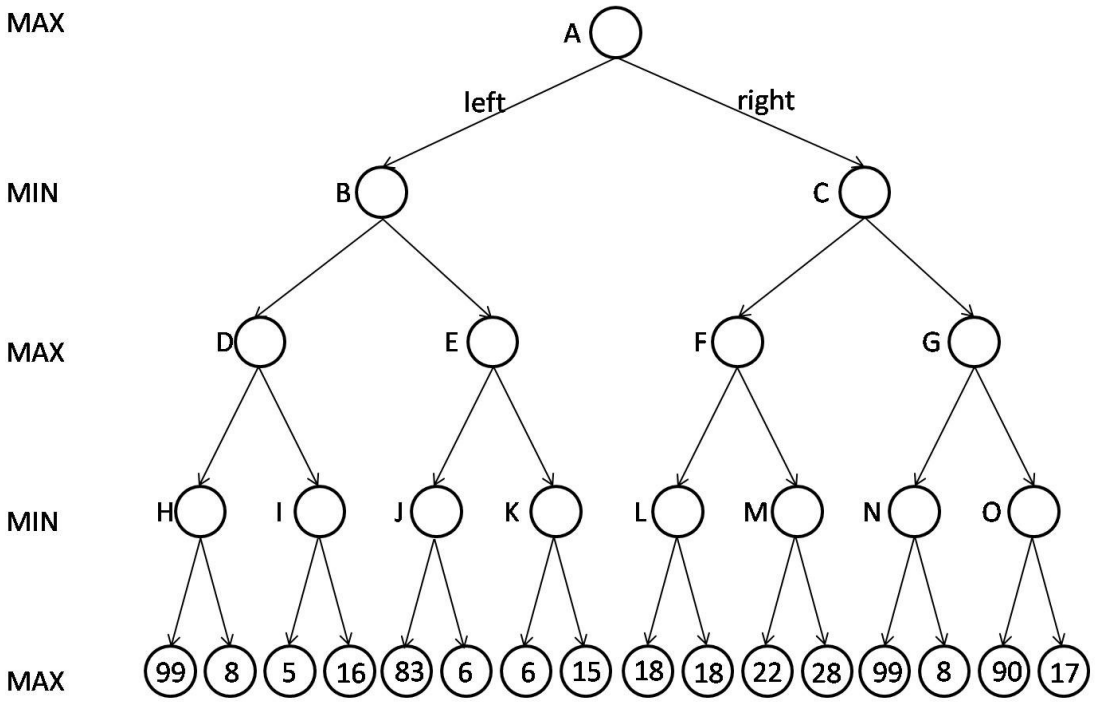

Figure 2: Fill in the alpha-beta pruning values and mark nodes that aren't visited.

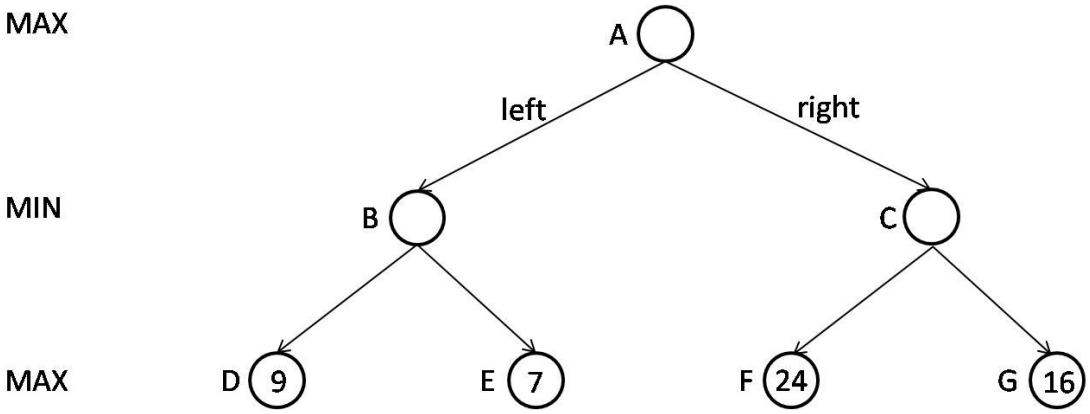

Figure 3: Minimax run to level 2.

0.3 Optimized alpha-beta pruning

In this question, we still consider the same game tree. But now MAX would like to increase the number of nodes not examined by the Alpha-Beta pruning algorithm. For that purpose, MAX first runs the Minimax algorithm down to depth 2, not to select a move, but only to acquire some information that may help the Alpha-Beta pruning algorithm. So, the tree generated at depth 2 by Minimax consists of nodes A, B, ..., F and G. Figure 3 shows this tree along with the values of e at the leaf nodes. Minimax backs-up these values at nodes B and C, but the backed-up values are not shown.

Now, MAX runs the Alpha-Beta pruning algorithm down to depth 4. Here, this algorithm uses both the values of e at depth 2 and the backed-up values computed by Minimax at nodes B and C to reorder the nodes of the depth-4 game tree at depths 1 and 2, with the goal to maximize the number of nodes that will not be examined. [Note that, after reordering the nodes, the "left" arc from a node may be on the right, and vice-versa.]

1. How should the Alpha-Beta pruning algorithm reorder the nodes, given the information generated by Minimax? To answer this question, assign the labels B, C, ..., N and O to the nodes of the tree if figure 4, assuming that the Alpha-Beta pruning algorithm will perform a depth-first traversal of the tree from left to right. In addition, inside each leaf node give the value of the evaluation function.

2. How many nodes will not be examined by the Alpha-Beta pruning algorithm?

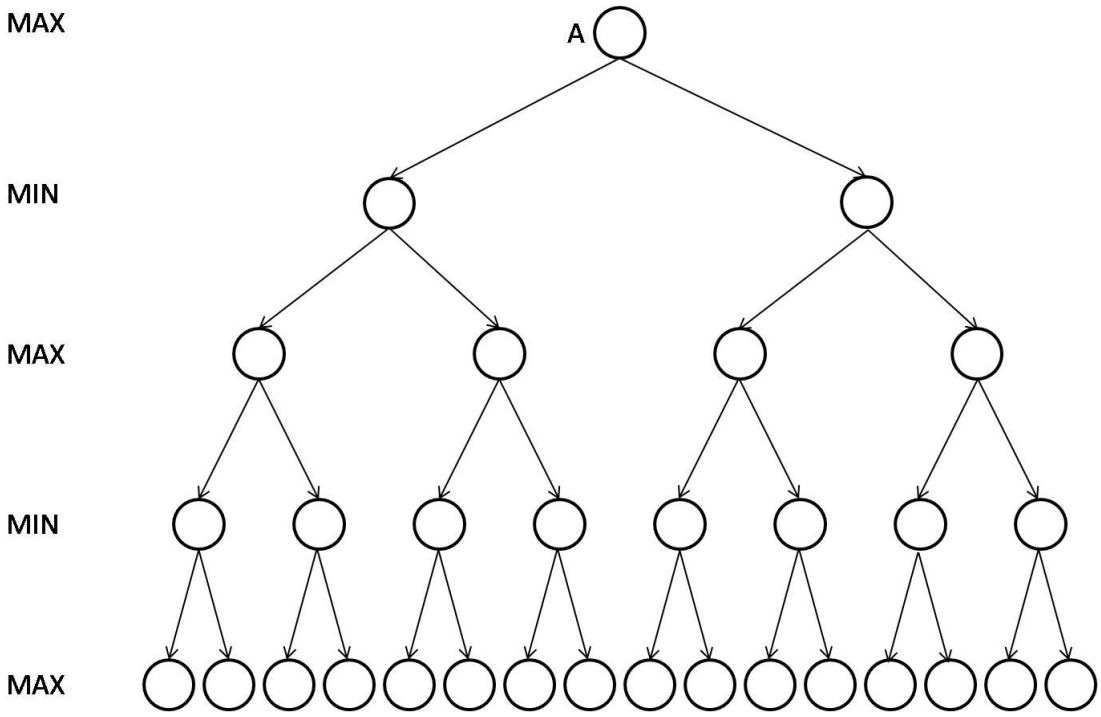

Figure 4: Fill in the letters to achieve maximum alpha-beta pruning.

0.4 Tic-Tac-Toe

[25 points]

This problem exercises the basic concepts of game playing, using tic-tac-toe (noughts and crosses) as an example. We define X_n as the number of rows, columns, or diagonals on the board with exactly n X's and no O's. Similarly, O_n is the number of rows, columns, or diagonals with just n O's. The utility function assigns $+1$ to any board with $X_3 = 1$ and -1 to any board with $O_3 = 1$. All other terminal boards have utility 0. For nonterminal boards, we use a linear evaluation function defined as:

$$Eval(s) = 3X_2(s) + X_1(s) - 3O_2(s) - O_1(s) \tag{1}$$

1. Approximately how many possible games of tic-tac-toe are there?
2. Show the whole game tree starting from an empty board down to depth 2 (i.e., one X and one O on the board), taking symmetry into account.
3. Mark on your tree the evaluations of all the positions at depth 2.
4. Using the minimax algorithm, mark on your tree the backed-up values for the positions at depths 1 and 0, and use those values to choose the best starting move.

5. Circle the nodes at depth 2 that would not be evaluated if alpha–beta pruning were applied, assuming the nodes are generated in the optimal order for alpha–beta pruning.

1 Knowledge Representation

1.1 Conversion to clause form

[10 points]
Convert the following statments to clause form (show your work):

1.

$$\forall x : [P_1(x) \wedge P_2(x, A)] \Rightarrow [P_3(x, B) \vee (\forall y : \exists z : P_3(y, z) \Rightarrow P_4(x, y))]$$

2.

$$(P \Rightarrow Q) \Rightarrow ((Q \Rightarrow R) \Rightarrow (P \Rightarrow R))$$

1.2 English $\rightarrow$ Logic

[10 points]
Translate the following sentences into propositional logic, making the meaning of your propositional variables clear.

1. Neither the storm blast nor the flood did any damage to the House.
2. Drivers should neither drive over 65 miles per hour nor cross the red light, or they will get a ticket.

1.3 Most general unifier

[15 points]
Determine whether the members of each of the following pairs of expressions unify with each other. If so, find the most general unifier (MGU); If not, give a brief explanation. Note – free variables are assumed to be universally quantified.

1. (a) $Color(Hat(Postman), Blue)$
   (b) $Color(Hat(y), x)$
2. (a) $R(F(y), y, x)$
   (b) $R(x, F(A), F(v))$
3. (a) $Loves(x, y)$
   (b) $Loves(y, x)$

1.4 Resolution

[15 points]

1. Given the clauses $\{p(a), q(a)\}, \{\neg p(x), r(x)\}, \{\neg q(a)\}$, use Resolution to derive the clause $\{r(a)\}$.
2. Given $\forall x\ p(x) \Rightarrow q(x)$, use Resolution to prove $\forall x\ p(x) \Rightarrow \forall x\ q(x)$.
3. Use Resolution to prove $\forall x\ (((p(x) \Rightarrow q(x)) \Rightarrow p(x)) \Rightarrow p(x))$.

Show your work.