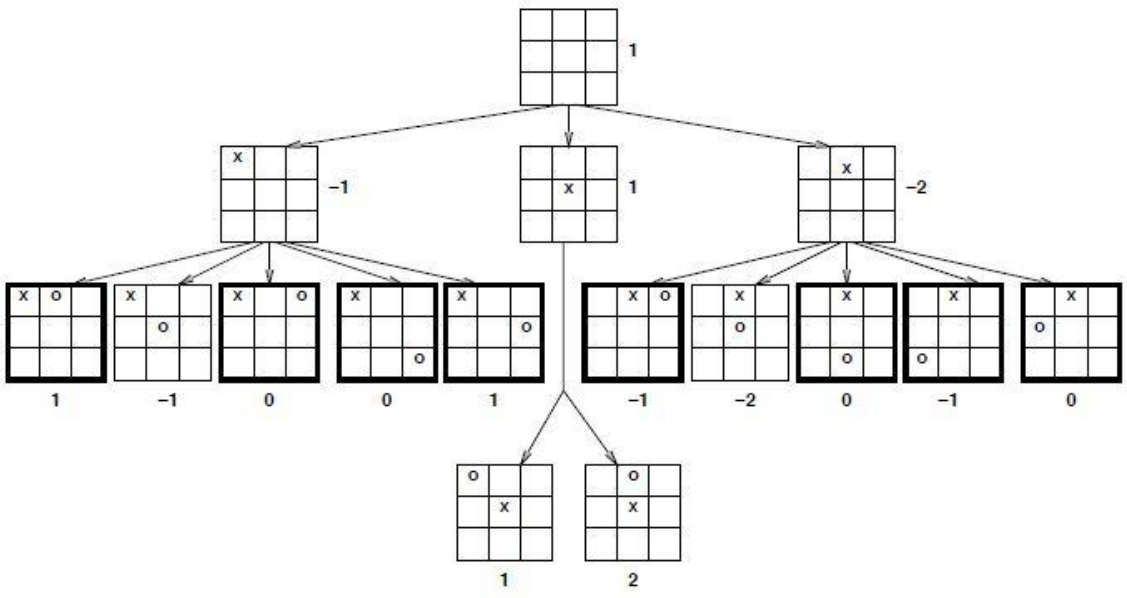
EN.520.650.01.SP22: Machine Intelligence
Homework - 2 Solutions

---

1. This problem exercises the basic concepts of game playing, using tic-tac-toe (noughts and crosses) as an example. We define Xn as the number of rows, columns, or diagonals with exactly n X's and no O's. Similarly, On is the number of rows, columns, or diagonals with just n O's. The utility function assigns +1 to any position with X3=1 and −1 to any position with O3=1. All other terminal positions have utility 0. For nonterminal positions, we use a linear evaluation function defined as

Eval(s)=3X2(s)+X1(s)−(3O2(s)+O1(s)). 1. Approximately how many 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 was applied, assuming the nodes are generated in the optimal order for alpha-beta pruning.

Solution:



For a, there are at most 9! games. (This is the number of move sequences that fill up the board, but many wins and losses end before the board is full.) For b-e, the figure

given below shows the game tree, with the evaluation function values below the terminal nodes and the backed-up values to the right of the non-terminal nodes. The values imply that the best starting move for X is to take the center. The terminal nodes with a bold outline are the ones that do not need to be evaluated, assuming the optimal ordering.

---

2. For each of the following assertions, say whether it is true or false and support your answer with examples or counterexamples where appropriate.
a. A hill-climbing algorithm that never visits states with a lower value (or higher cost) is guaranteed to find the optimal solution if given enough time to find a solution.
b. For any local-search problem, hill-climbing will return a global optimum if the algorithm is run starting at any state that is a neighbor of a globally optimal state.
c. Stochastic hill climbing is guaranteed to arrive at a global optimum.
d. In a continuous space, gradient descent with a fixed step size is guaranteed to converge to a local or global optimum.
e. Gradient descent finds the global optimum from any starting point if and only if the function is convex

Solutions

a. False. Such an algorithm will reach a local optimum and stop (or wander on a plateau)

b. False. The intervening neighbor could be a local minimum and the current state is on another slope leading to a local maximum. Consider, for example, starting at the state valued 5 in the linear sequence 7, 6, 5, 0, 9, 0, 0.

c. False. Stochastic hill-climbing chooses from the uphill moves at random, but, unlike simulated annealing, it stops if there are none, so it still gets trapped in local optima.

d. False. While convexity implies that gradient descent will find a global optimum, the converse is not true. A function can have no local optima without being convex—e.g., in one dimension, it just needs to be strictly decreasing/increasing to the left/right of the global optimum.

e. True. Suppose f is convex and x_0 is a starting point. Then applying gradient descent, we obtain the local minimum of f. Since f is convex, it is also global optimum. Since x_0 is arbitrary, proved.

---

3. Mom, Dad, Baby, Student, Teacher, and Guide are lining up next to each other in six linear spots labeled 1 to 6, one to a spot. Baby needs to line up between Mom and Dad. Student and Teacher need to be next to each other. Guide needs to be at one end, in spot 1 or 6. Formulate this problem as a CSP: list the variables, their domains, and the constraints. Encode unary constraints as a constraint rather than pruning the domain. (No need to solve the problem, just provide variables, domains, and constraints.)

Solution

Variables: {M, D, B, S, T, G}

Domains: {1, 2, 3, 4, 5, 6} for all variables

Constraints: alldiff(M, D, B, S, T, G), $|B - M| = 1$, $|B - P| = 1$, $|S - T| = 1$, $G \in \{1, 6\}$

---

4. In a full-depth minimax search of a tree with depth D and branching factor B, with α−β pruning, what is the minimum number of leaves that must be explored to compute the best move?

Solution:

There are B^D leaf nodes. In α − β pruning we have to look at all the first player's moves but we only have to look at one of the second player's moves. Thus, we skip every other level and only have to look at $B^{D/2}$.

---

5. Which of the following statements about alpha-beta pruning are true or false?
a. Alpha-beta pruning may find an approximately optimal strategy, rather than the minimax optimal strategy.
b. Alpha-beta prunes the same number of subtrees regardless of the order of child nodes.
c. Alpha-beta generally requires more run-time than minimax on the same game tree.

Solution: None of these are true. Alpha-beta will always find the optimal strategy against an opponent that plays optimally. If an ordering heuristic is available, we can expand nodes in an order that maximizes pruning. Alpha-beta will require less run-time than minimax except in contrived cases.

---

6. Ali, Bo, Cleo, and Dallas are picking their entrees at a restaurant. The choices are pasta, quesadillas, risotto, and sushi. They have some strict dietary preferences:

• Cleo will not order sushi.

• Ali and Bo want to steal each other's food, so they will order different dishes.

• Bo likes carbs, so he will only order pasta or risotto.

• Cleo likes to be unique in her food orders and will not order the same dish as anybody else, with one exception: Ali and Cleo are actually twins, and always order the same dish as each other.

• Dallas really dislikes quesadillas and will not order them.

Answer the following questions for this situation:

a. If we formulate this as a CSP with variables {A, B, C, D}, each with domain {P, Q, R, S}, what are the constraints?

b. We will run a basic backtracking search to solve this CSP and make sure that every person (variable) is matched with their dream dish (value). We will select unassigned variables in alphabetical order, and we will also iterate overvalues in alphabetical order. What assignment will backtracking search return?

c. Now assume that no values have been assigned to the variables and we will run one iteration of the forward checking. What value(s) will be eliminated for which variables if we assign "pasta" to Ali?

d. Now assume we will solve the problem with a local search using the min-conflicts algorithm. Assume we start with the initial assignment {A = P, B = P, C = P, D = P} and choose B as the variable to change. How many conflicts does the current value of B pose? What value for B would minimize the number of conflicts?

Solution

a. $C \neq S$; $A \neq B$; $B \in \{P, R\}$, $C \notin \{B, D\}$; $D \neq Q$.
b. {A = P; B = R; C = P; D = R}.
c. Eliminate P from B; eliminate Q; S;R from C; eliminate nothing from D.

d. B = S conflicts with A ≠ B;B ∈ {P, R}, C ∉ {B, D}, so 3 conflicts. Setting B = R would resolve all 3 of B's conflicts; no other value choice would do that.
