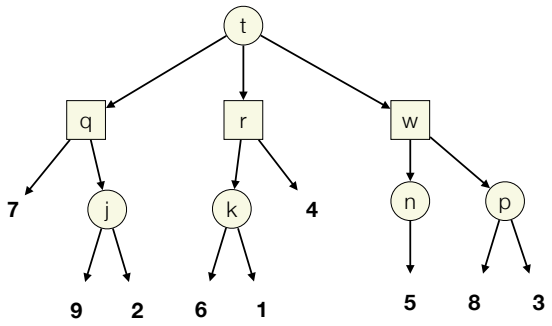
Unless otherwise indicated, assume that the root player is a MAX player and leaf nodes are labelled with the MAX player's score.

1. Consider the following state graph

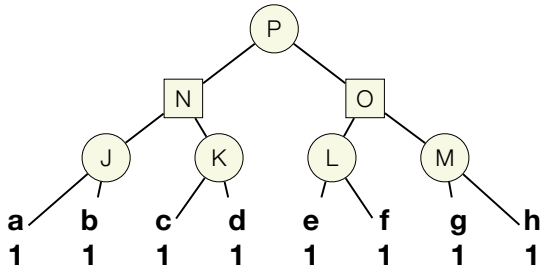


   (a) What are the minimax values for each nonterminal node?

   (b) What are the negamax values for each nonterminal node? Assume that each terminal node is labelled with the score for the player-to-move.

   (c) Why are these two values different?

2. Suppose we execute alpha-beta search on the following tree.



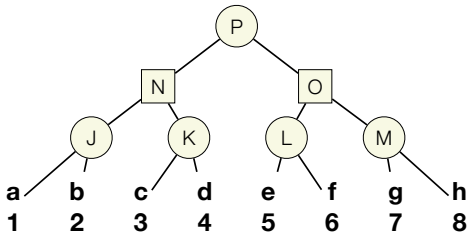
   (a) If we use the $\alpha \geq \beta$ pruning rule, which leaf nodes will be evaluated? How many nodes are pruned?

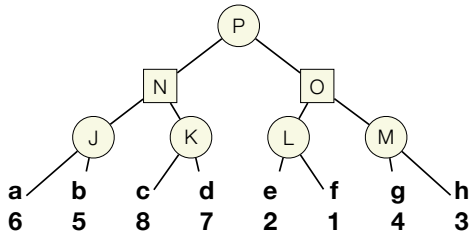
(b) *After* alpha-beta completes using the $\alpha \geq \beta$ pruning rule, what is known about the minimax value for each nonterminal node? E.g., use ≥ 3 to indicate a lower bound of 3, ≤ 2 to indicate an upper bound of 2, 7 to indicate exactly 7, and ? to indicate that nothing is known.

(c) If we use the $\alpha > \beta$ pruning rule, which leaf nodes will be evaluated? How many nodes are pruned?

(d) *After* alpha-beta completes using the $\alpha > \beta$ pruning rule, what is known about the minimax value for each nonterminal node?

3. Suppose we execute alpha-beta search on the following two trees using the $\alpha \geq \beta$ pruning rule.





(a) Which leaf nodes will be evaluated in the left tree? How many nodes are pruned?

(b) *After* alpha-beta completes in the left tree, what is known about the minimax value for each nonterminal node?

(c) Which leaf nodes will be evaluated in the right tree? How many nodes are pruned?

(d) *After* alpha-beta completes in the right tree, what is known about the minimax value for each nonterminal node?

4. Consider the follow code from `abeta/negamax.py`, with three missing expressions:

```
def negamax(d, T, V, v): # leaf scores for player-to-move
    if isTerminalNode(v,V):
        val = V[v]
        return val
    val = _____expression_(A)_____
    for c in T[v]: # for each child c of v
        nmx = negamax(d+1, T, V, c)
        val = max(_____expression_(B)_____, _____expression_(C)_____)
    return val
```

(a) What should expression (A) be?

(b) What should expression (B) and expression (C) be?

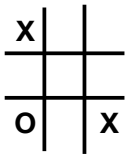
5. Consider the following code in `tt.py`:

```
Isos = ( (0,1,2,3,4,5,6,7,8),
 (0,3,6,1,4,7,2,5,8),
 (2,1,0,5,4,3,8,7,6),
 (2,5,8,1,4,7,0,3,6),
 (8,7,6,5,4,3,2,1,0),
 (8,5,2,7,4,1,6,3,0),
 (6,7,8,3,4,5,0,1,2),
 (6,3,0,7,4,1,8,5,2)  )
```

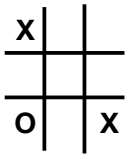
(a) What is the purpose of this code?

(b) What is the result of applying the last transformation in `Isos` to the position `(1,0,0,2,1,0,0,0,0)`?

6. Give a proof tree that proves that the following position is a win for X.



7. What are the non-isomorphic children of the following position? Use the canonical representation from lecture that chooses the smallest numerical representation of a position, where empty cells map to 0, X cells map to 1, and O cells map to 2.

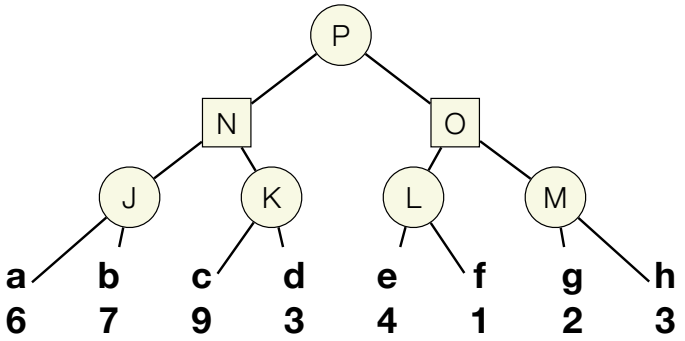


8. Consider the Nim position `(7,7,7,7,7,7,7)`:

   (a) How many children does this position have? *There is no need to list the children.*

   (b) How many *non-isomorphic* children does this position have? List them all in canonical form.

9. This alpha-beta search has just reached K. In the table below, list the new values of `so_far`, `alpha`, and `beta` in order, every time at least one of them changes. We've listed the changes up to but not including node K.



chg	node	so_far	alpha	beta
1	P	$-\infty$	$-\infty$	∞
2	N	∞	$-\infty$	∞
3	J	$-\infty$	$-\infty$	∞
4	J	6	6	∞
5	J	7	7	∞
6	N	7	$-\infty$	7
7				
8				
9				
10				
11				
12				
13				