Artificial Intelligence - 2018
Adversarial Search

Prepared By;
Vijani Supeshala Piyawardana | BSc (Hons) Computer Science |
Demonstrator - COMP3004L, COMP3008L - UCD |
NSBM Green University Town

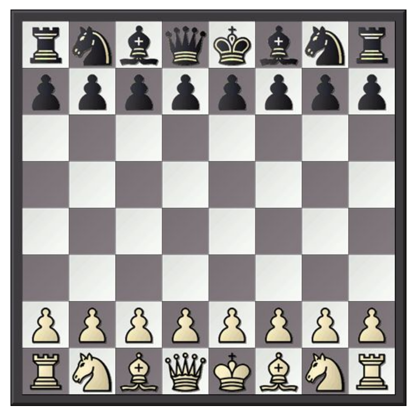
Adversarial Search

Introduction

1. Trying to win, when someone else is actively trying to beat us.
   Involves techniques similar to searching, outright use of search techniques but the adversary is another variable to take care of.
   Ex: games

2. Often considerably widens the search space.
   Huge search space, visualized by a search tree.
   Ex: chess game

   - Average branching factor : 35
   - Game running for ~50 moves
   - 10^{154} nodes (10^{40} distinct ones)
     - Impossible to search them all
   - Have to act within a time limit



3. Player has only control on moves on alternative levels on the search tree, adversary moves in other cases.

4. We can't predict how adversary play, but can consider all its moves and compute most favourable.

5. A common assumption is :- adversary will do the best move for them. It is a variable to take care of.

Min Max algorithm in Adversarial Search

- Two players, "Max vs Min" used in zero sum games of perfect information.

  Zero sum games : one player's gain (or lose) of utility, is balanced by the loses (or gains) of the utility of the other player. Total gains - Total loses = zero. So, strictly competitive.

  Perfect information : nothing to hide

- Search strategy id MinMax strategy.

  MinMax strategy : choose best move assuming the opponent plays optimally.

- MinMax algorithm is a recursive algorithm to choose the next move (in position with highest minmax value) in an n-player game. (here we are talking about two-player game)

> The MinMax Algorithm
>
> 1. Recursively calculate the minmax value of the possible choices at the current state, then pick the best.
> 2. The recursion implicitly implements a depth first search.
> 3. Optimal
> 4. Complexity is O(b^m) ; b - branching factor, m - no of levels

- A value is associated to find the optimal move.
- In a search tree, of a two-player game, two kinds of nodes :-
  - Representing player moves : max nodes

    □ or △

  - Representing adversary's moves : min nodes

    ○ or ▽

- Max node's goal

  To MAXIMIZE the value of the sub-tree rooted at the node.
  To do this, a MAX node chooses the child with the greatest value, which becomes the value of the MAX node.

- Min node's goal

  To MINIMIZE the value of the sub-tree rooted at the node.
  To do this, a MIN node chooses the child with the least value,
  which becomes the value of the MIN node.

Minmax value

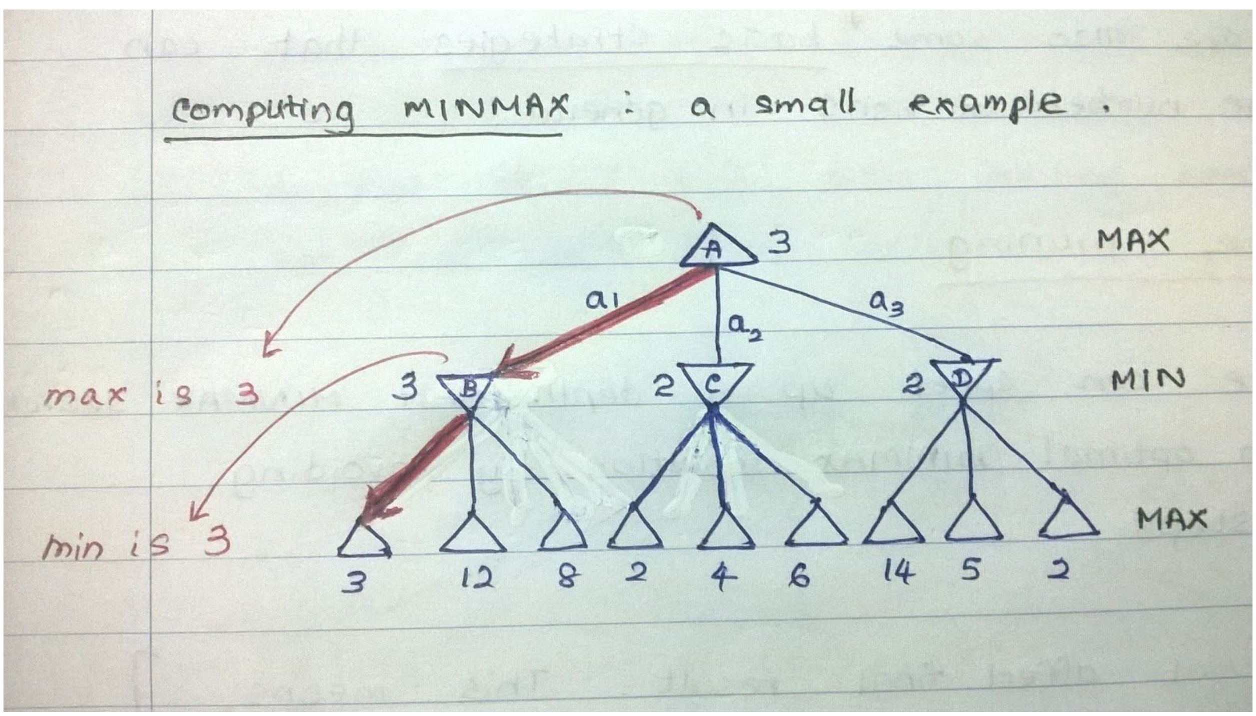
A game ends with a certain utility.

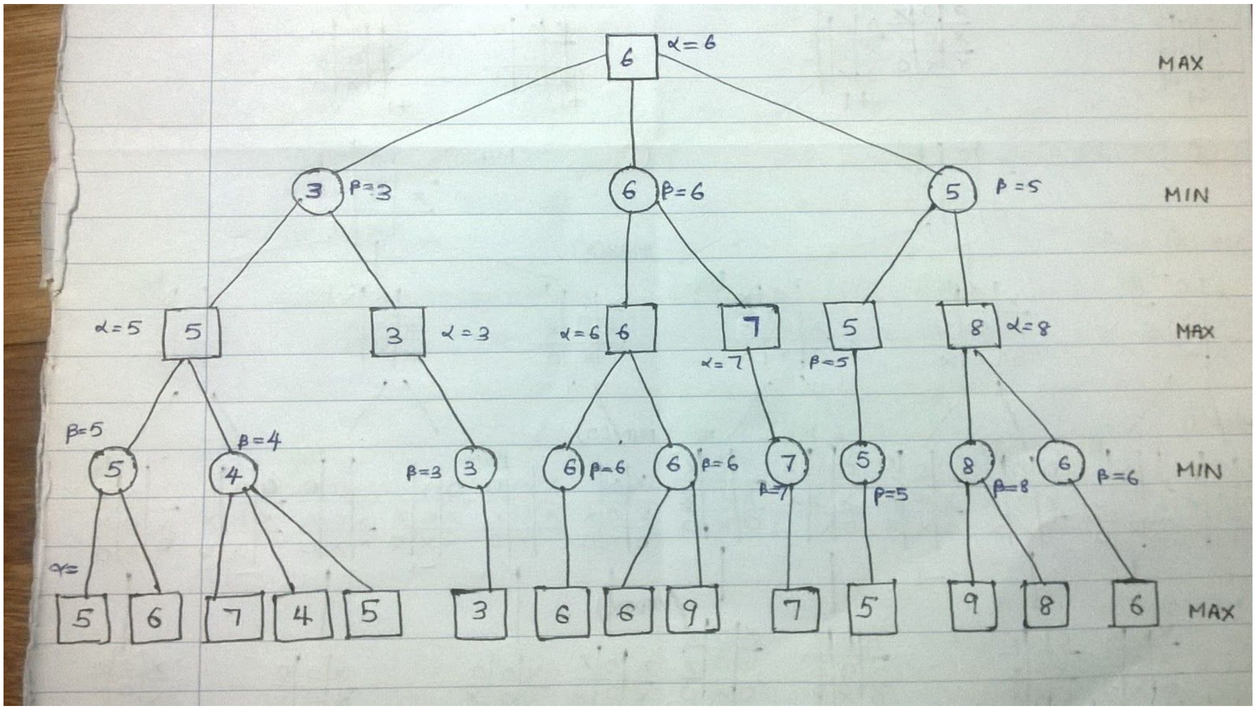
Utility : utility of a node is represented by minmax value.
+1 win
0 draw
-1 lose

Player who do the first move is MAX : will maximize the utility.
Adversary is MIN : will minimize the utility.

Minmax value is the utility for MAX of the final state we will end up in if both players play optimally.
It can be shown that if MIN plays sub-optimally, MAX will do even better,

Computing Minmax





Multi Player games

- Minmax can be extended to multiplayer games by making utility into a vector.
- However, multiplayer games can be much more complicated.
  - Can be alliances between players, that disrupt the normal purely selfish maximum utility behaviour.

Pruning

- In general, we cannot explore the whole tree, so we will need to prune it.
- That means, we will not need to examine whole branches of the tree, to reduce its' effective branching factor.
- For real games, pruning will require intelligent heuristics.
- But there are also some basic strategies that can cut down the number of visits in general.
- Pruning does not affect the final result. This means that, it gets the exact same result as does the full Minmax.

Alpha - Beta Pruning

This procedure can speed up a depth-first Minimax search and find an optimal minmax solution by avoiding unnecessary steps.

General principle of alpha-beta pruning

1. Consider a node “n”, somewhere in the tree, that Player has a choice of moving to that node or prune it.
2. If Player has a better choice “m” , either at the parent node of “n” or at any choice further up, then “n” will never be reached in actual play.
3. So, once we have found out enough about n (by examining some of its descendants) to reach this conclusion, we can prune it.

Value **V** : evaluation value of a node

Alpha **α** : highest value seen so far on MAX level

Beta **β** : lowest value seen so far on MIN level

If **$\alpha \geq \beta$** then prune

α-β pruning stops searching down a branch of a tree, when you can determine that it is a dead end/ reached to the leaf nodes of the tree.

Further explanation of α-β pruning

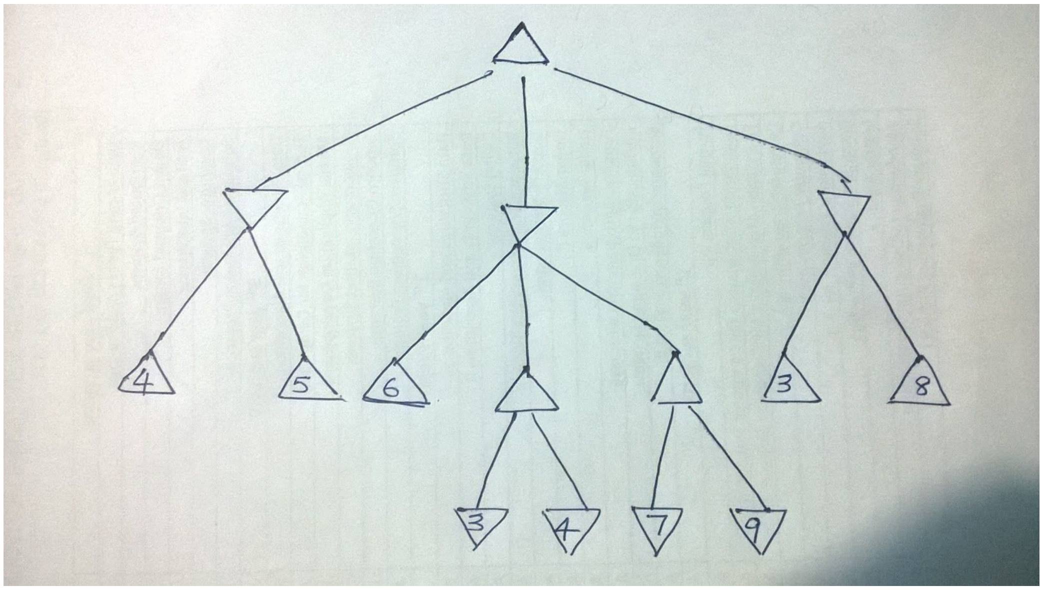
- When maximizing:

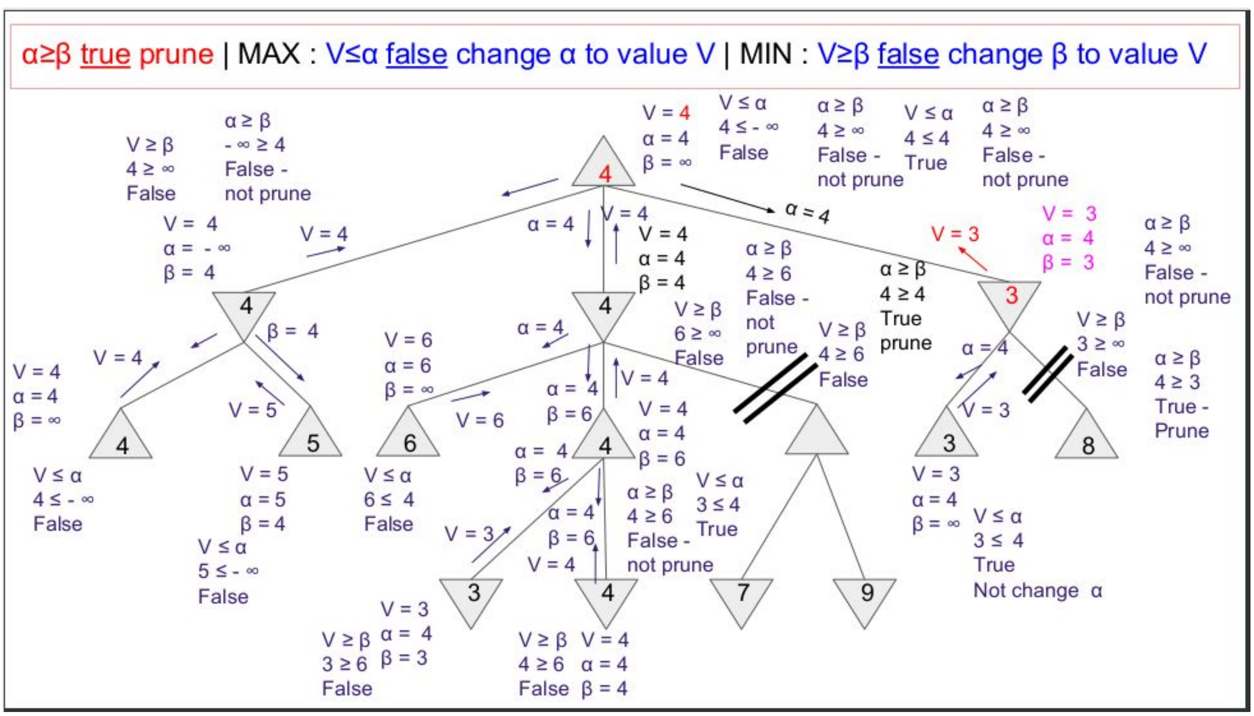
  Do not expand any more sibling nodes once a node has been seen whose evaluation is smaller than **α**
- When minimizing:

  Do not expand any more sibling nodes once a node has been seen whose evaluation is greater than **β**

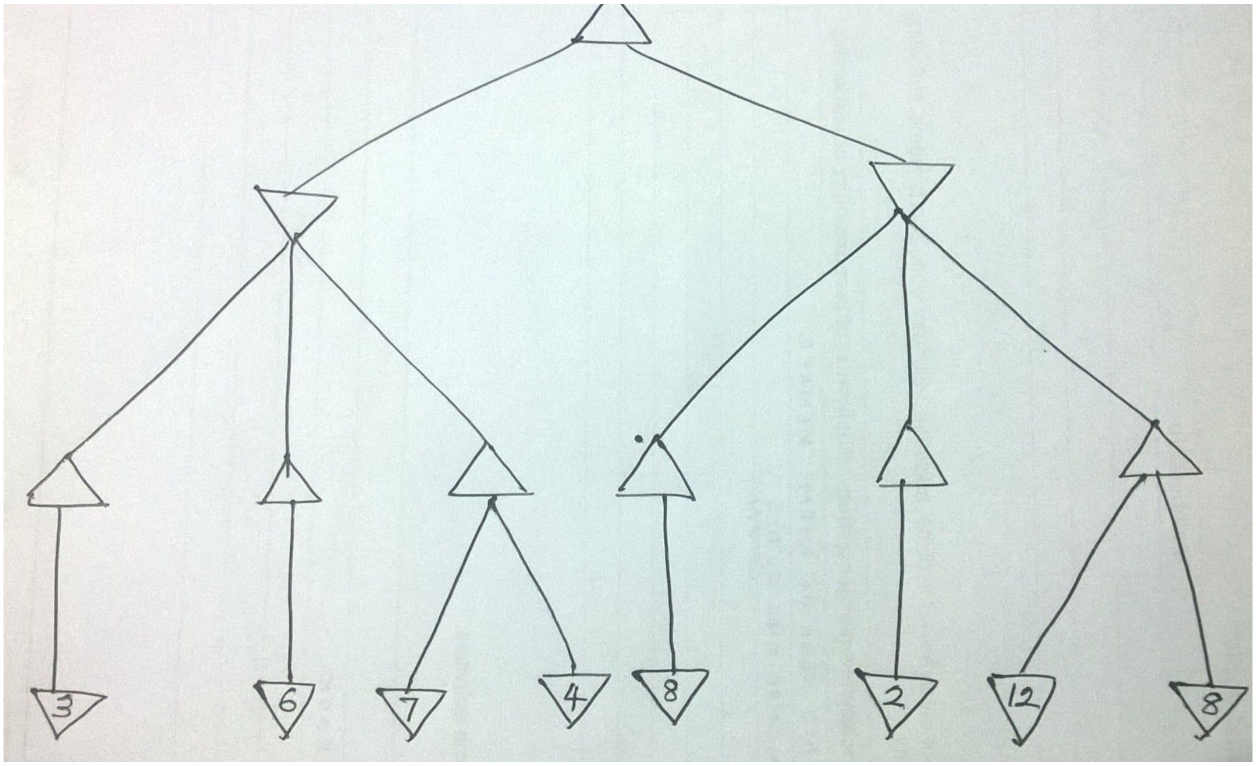
Some examples

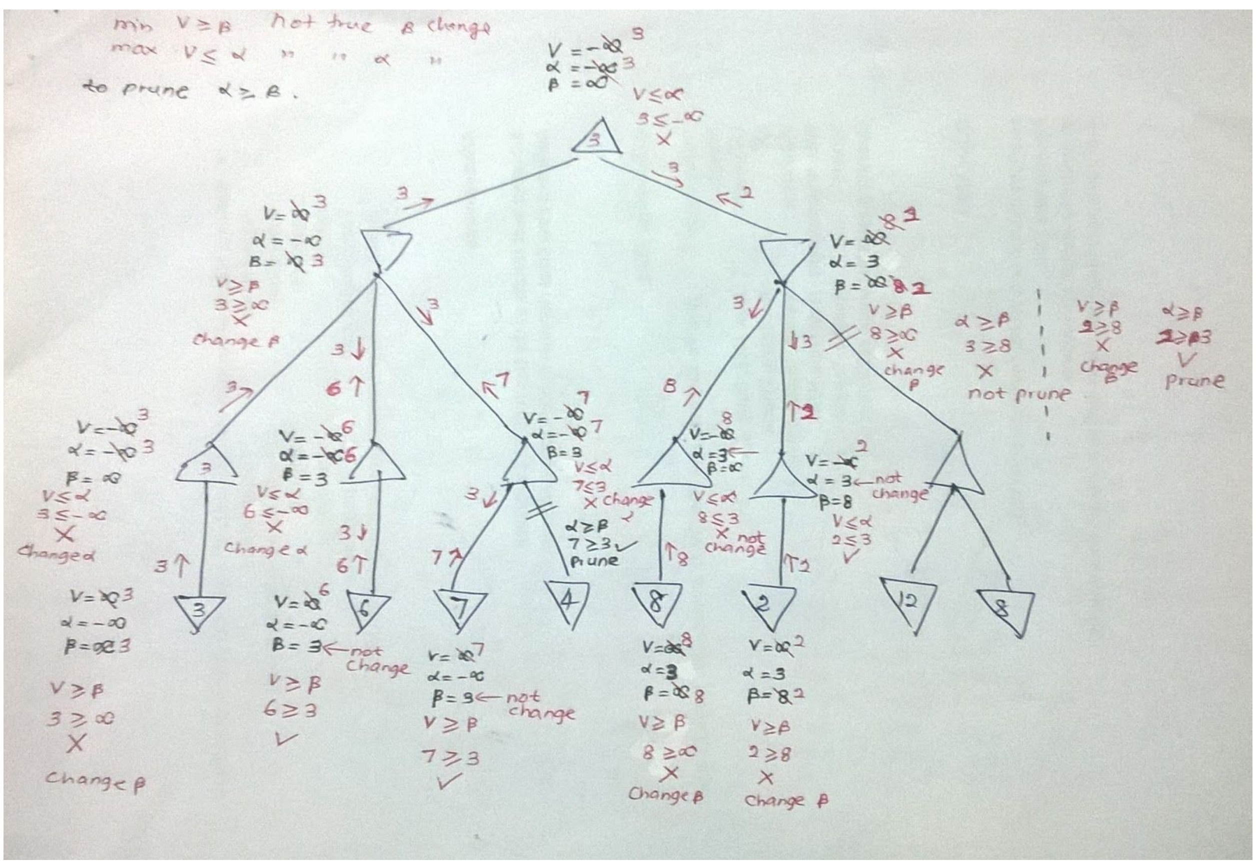
Example 1



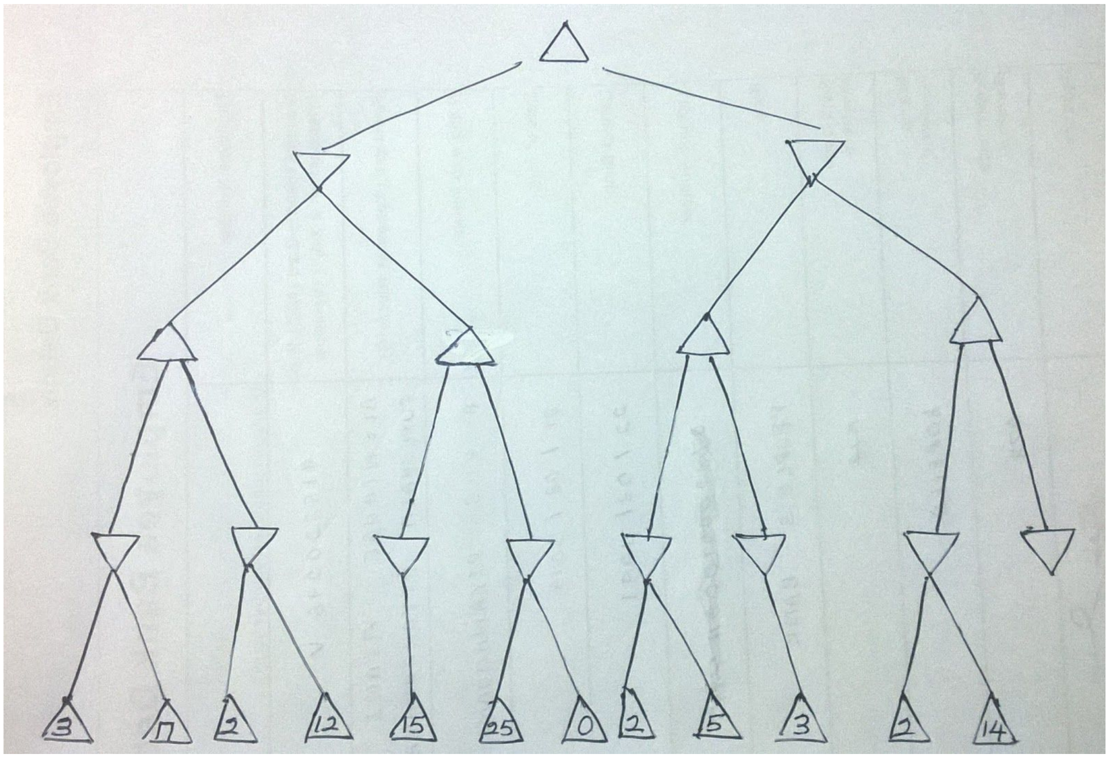


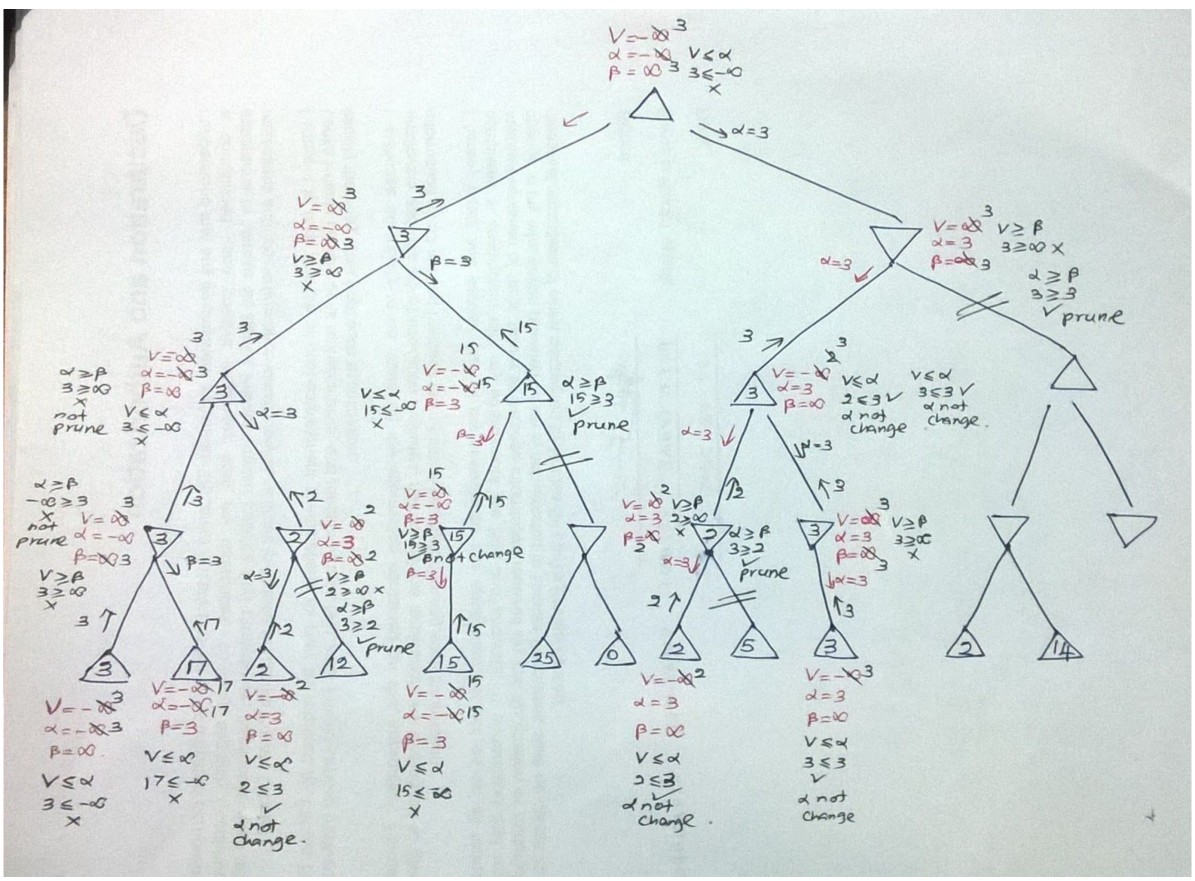
Example 2


3
6
7
4
8
2
12
8

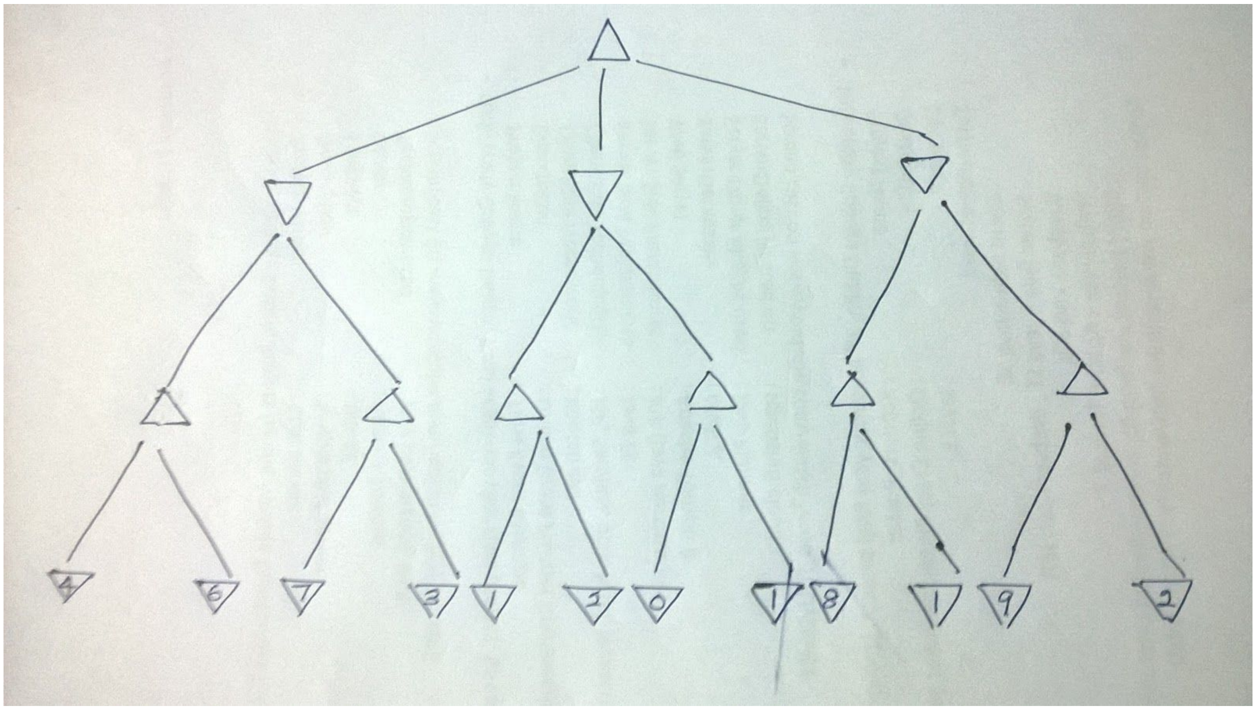

min V ≥ β not true β change
max V ≤ α " " α "
to prune α ≥ β.
not prune
Prune
change β
Changed
Change α
not change
Prune

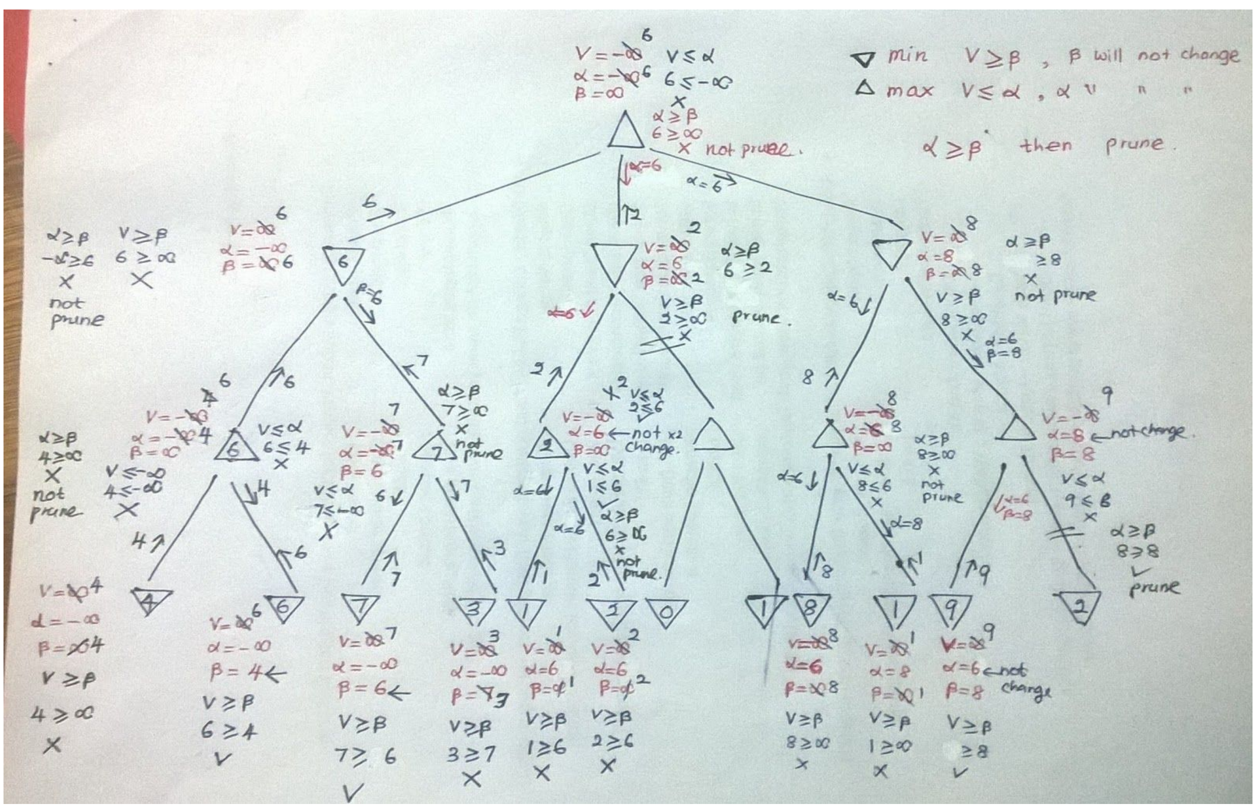
Example 3


3
17
2
12
15
25
0
2
5
3
2
14

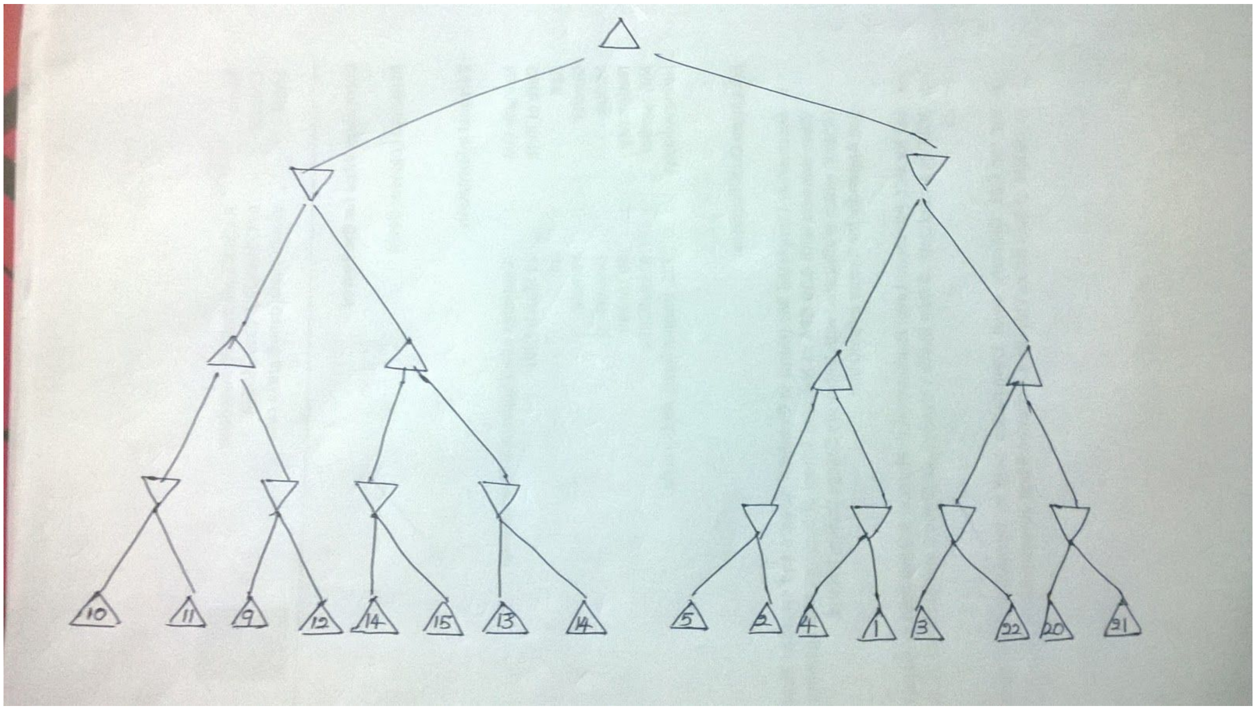

α=3
β=3
prune
prune
prune
prune
α not change
α not change
α not change
α not change
β not change
3
17
2
12
15
25
0
2
5
3
2
14

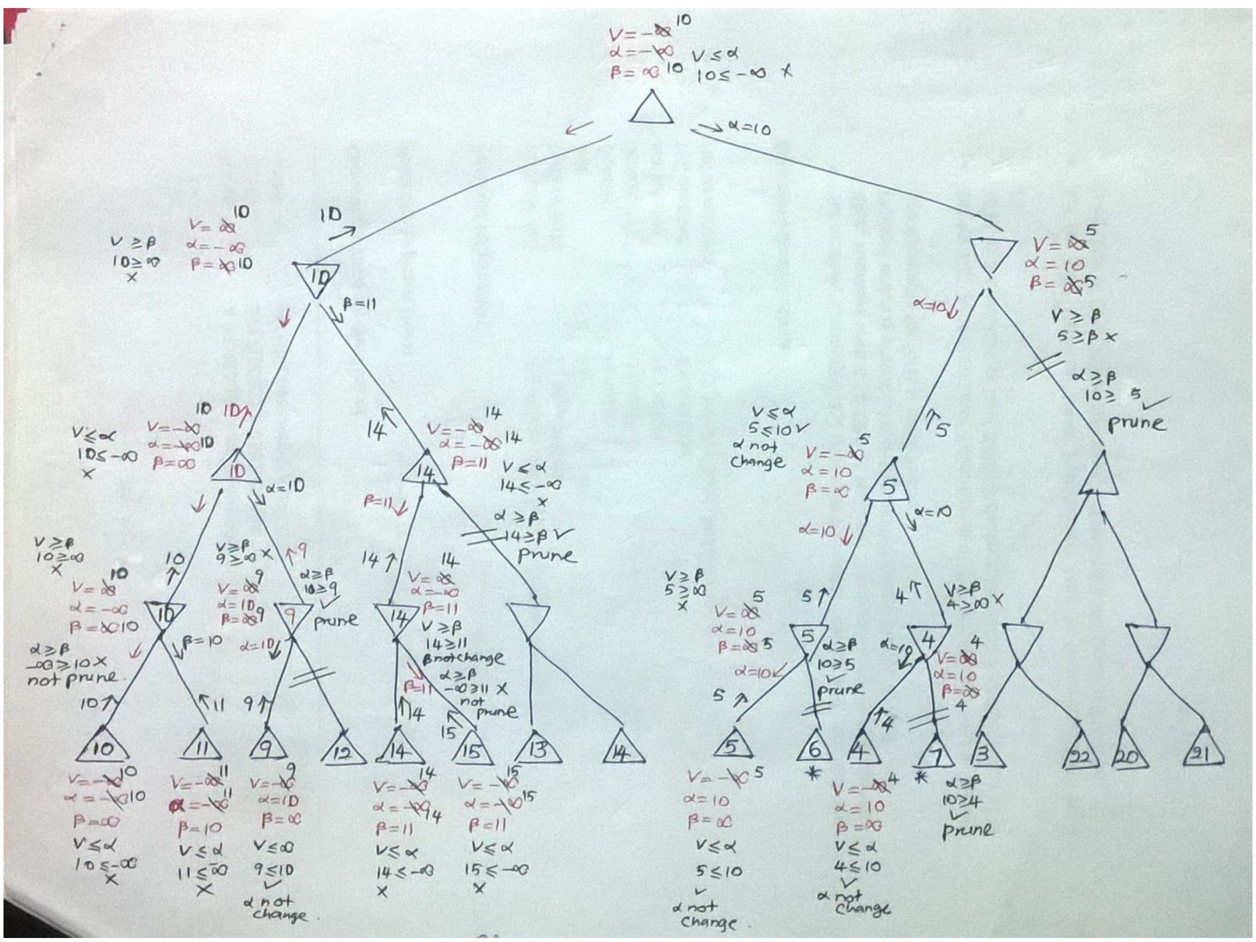
Example 4



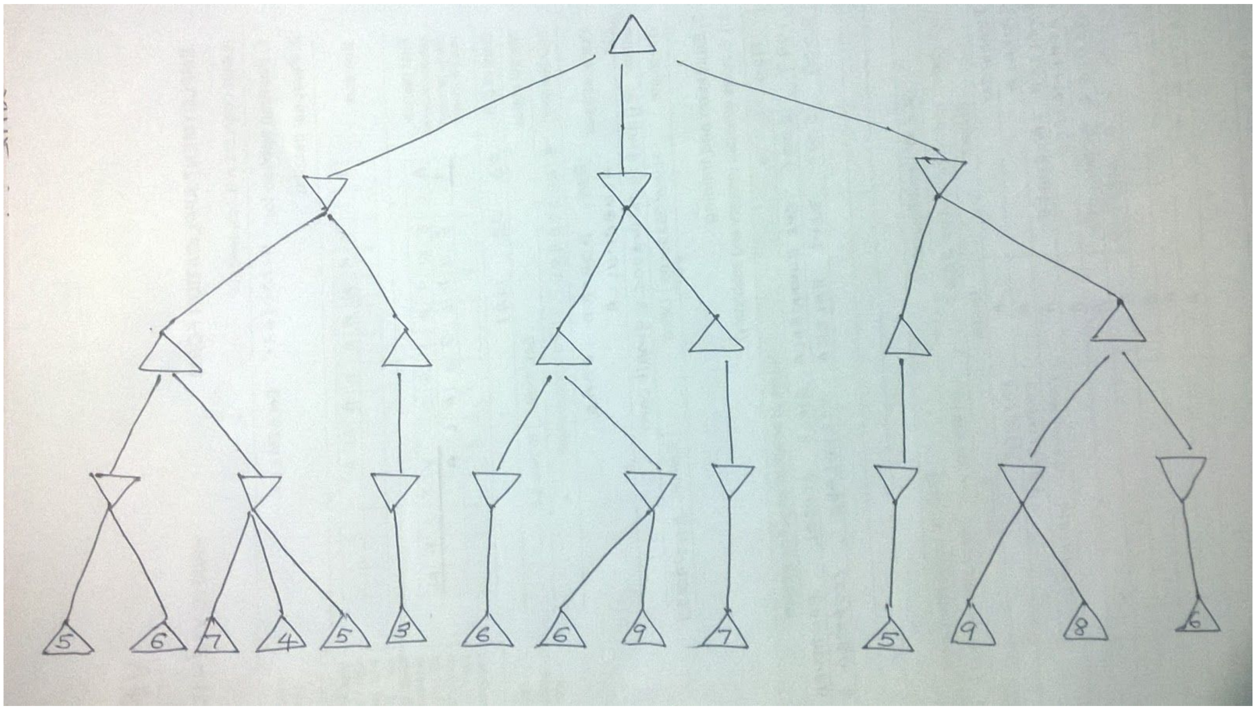


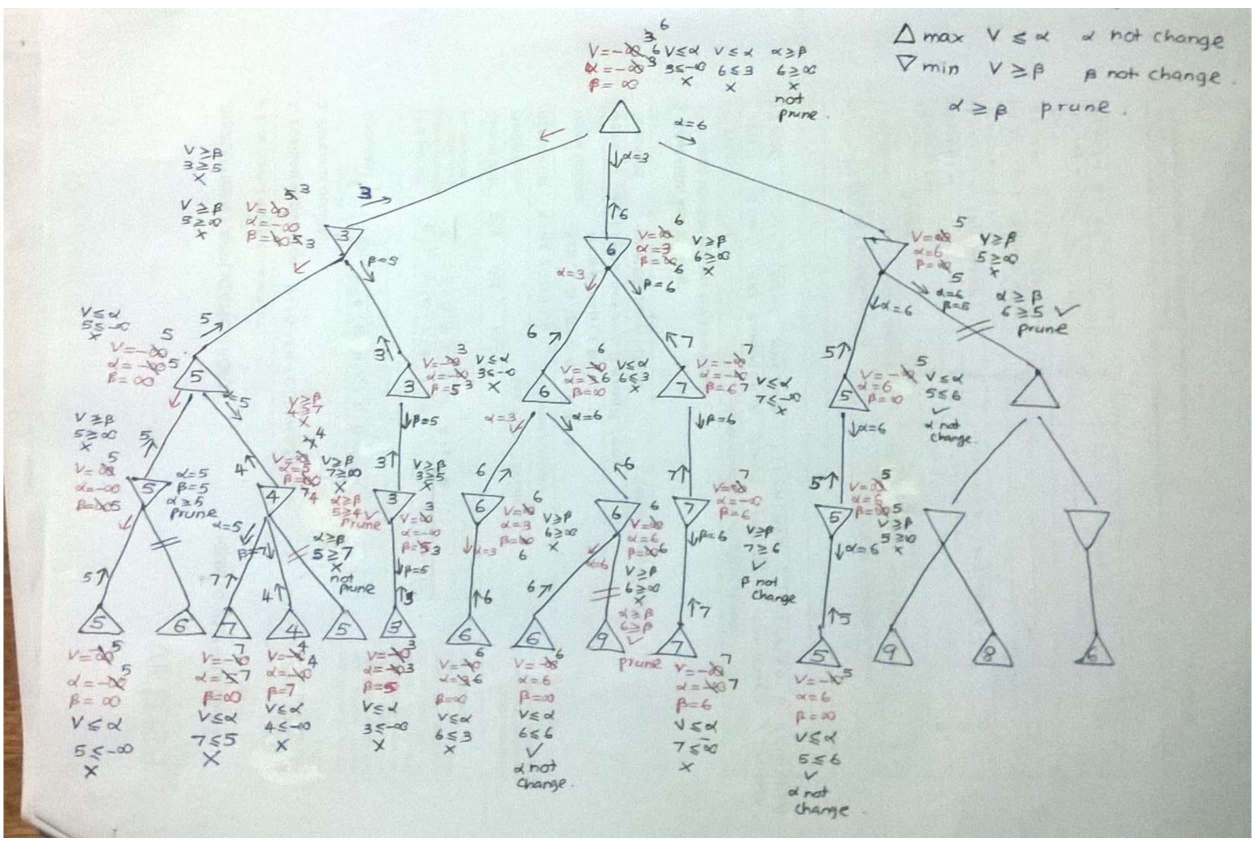
Example 5



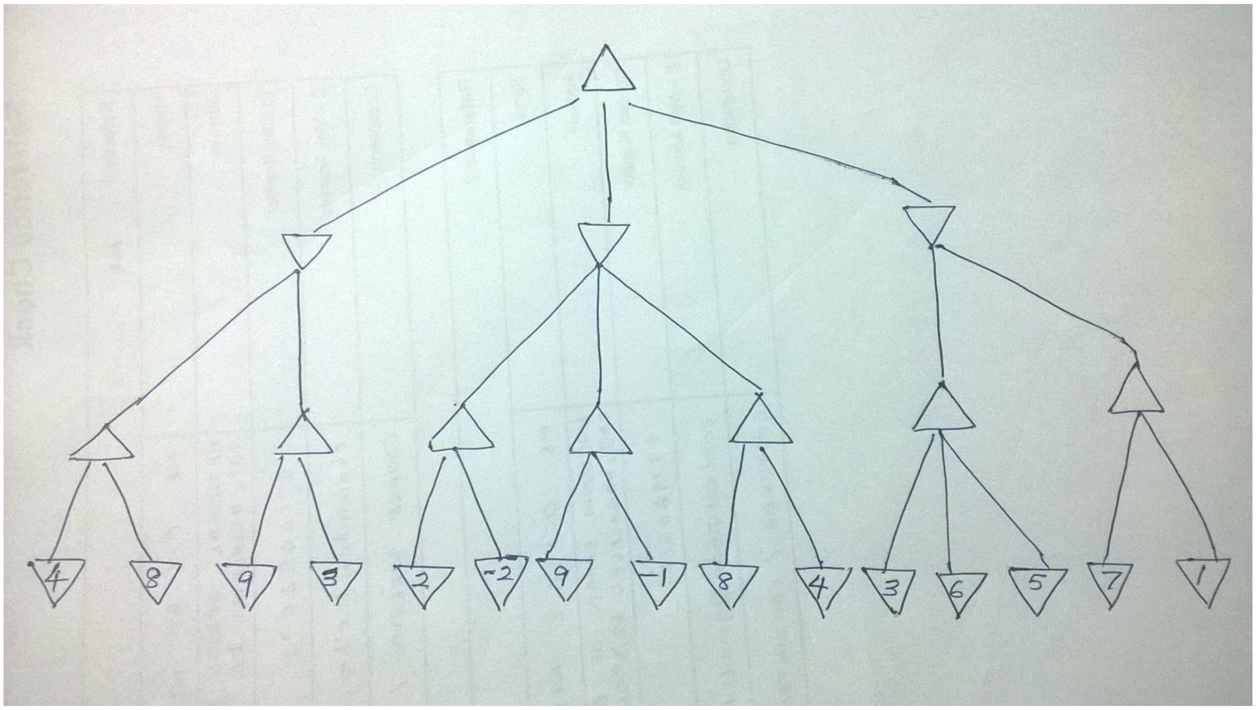


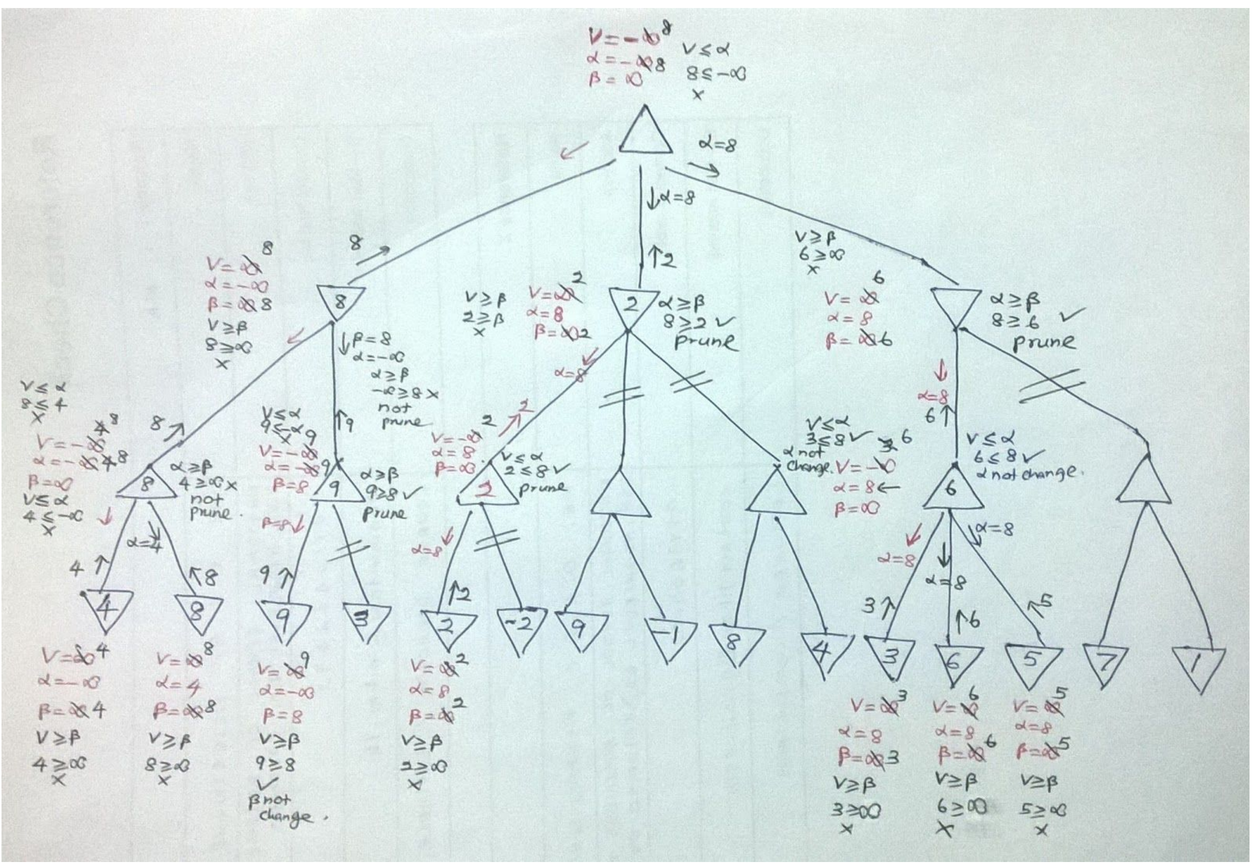
Example 6





Example 7


4
8
9
3
2
-2
9
-1
8
4
3
6
5
7
1


α=8
prune
not prune
α not change
β not change

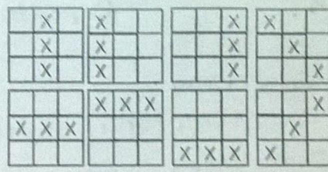
Exam Questions: Tic Tac Toe : 2014 Exam Question 2

Question 2 (a) What is meant by *adversarial* search? **(2 marks)**

(b) Describe the **minimax ~~algorithm~~** algorithm for adversarial search. **(4 marks)**

(c) How is **alpha-beta pruning** an improvement on minimax search. **(4 marks)**

(d) The game of noughts and crosses is a two-player game, played with pen and paper on a 3×3 grid. The first player chooses a grid position and places an X in that position. Player 2 then responds by placing O in one of the remaining grid spaces. Play continues in this manner. The game is over and Player 1 wins if three X's appear in either a horizontal, vertical or diagonal direction. The game ends with Player 2 winning, if three O's appear in either a horizontal, vertical or diagonal direction. If all 9 grid spaces are filled without either player winning, then the game is a draw. The winning configurations for X are shown below:



Consider the game in the following state, where it is Player 2's turn to move.

	O	
X	O	X
	X	

i. Draw the game tree for this game starting from this state. **(6 marks)**

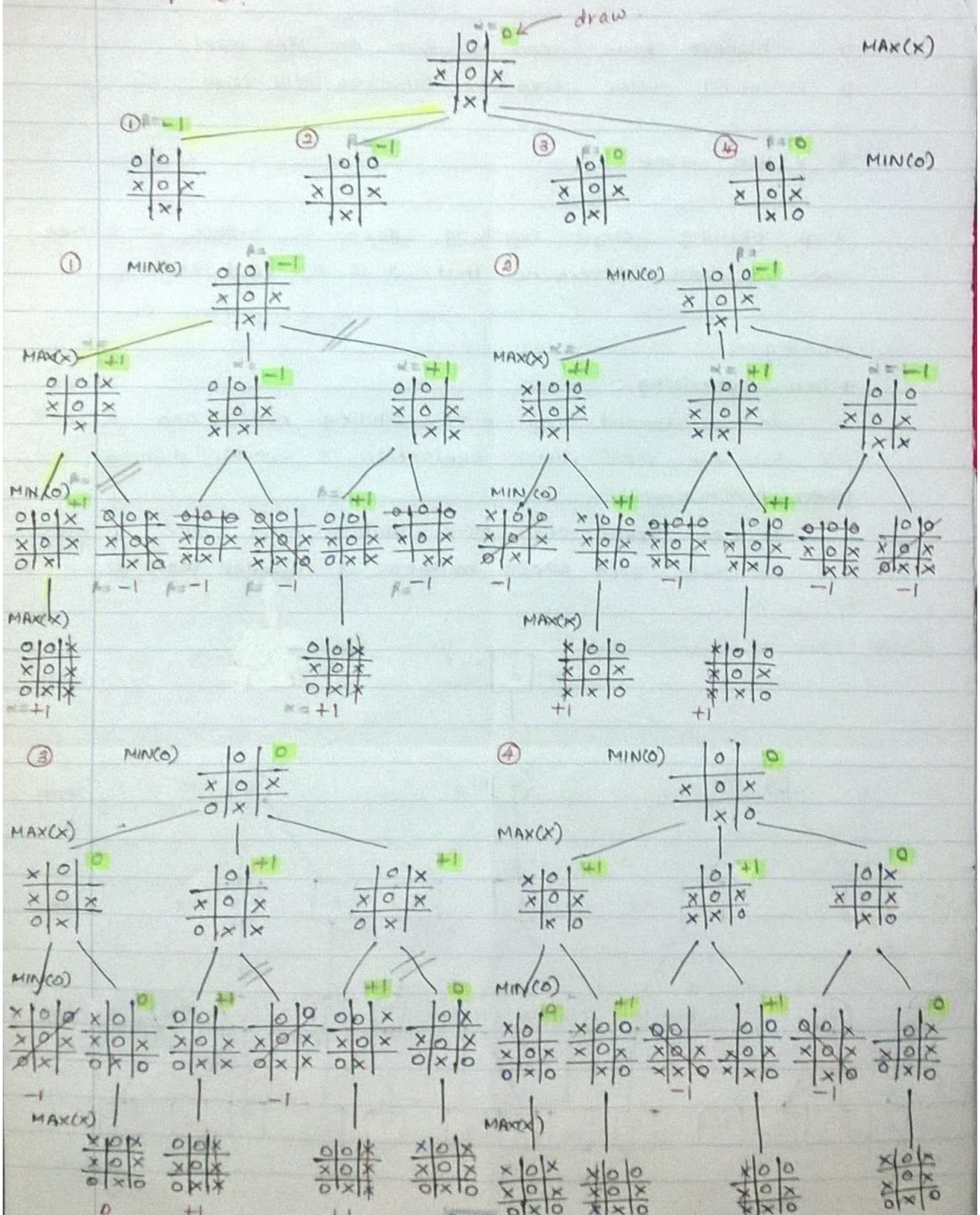
ii. Use the game tree and the minimax algorithm to show the outcome for each of the four possible moves, assuming each player plays the best move on each turn. **(4 marks)** batch 2 paper – Q2(d)

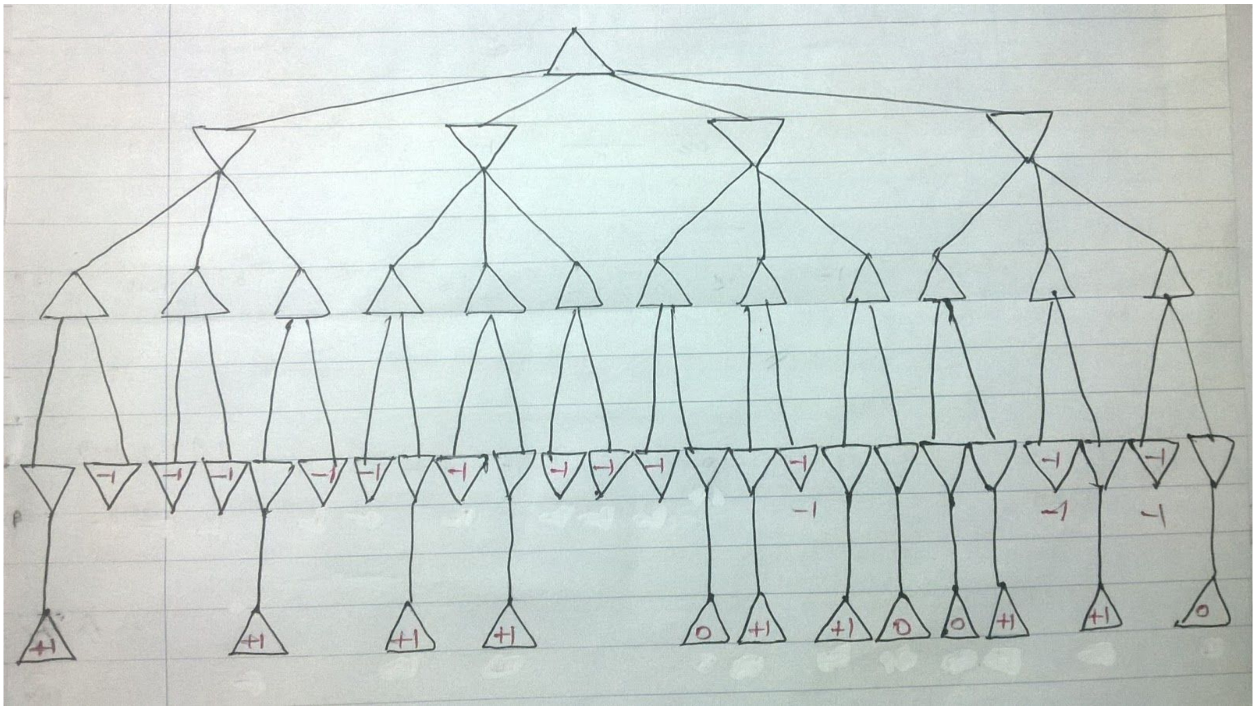
P1 – X
P2 – O

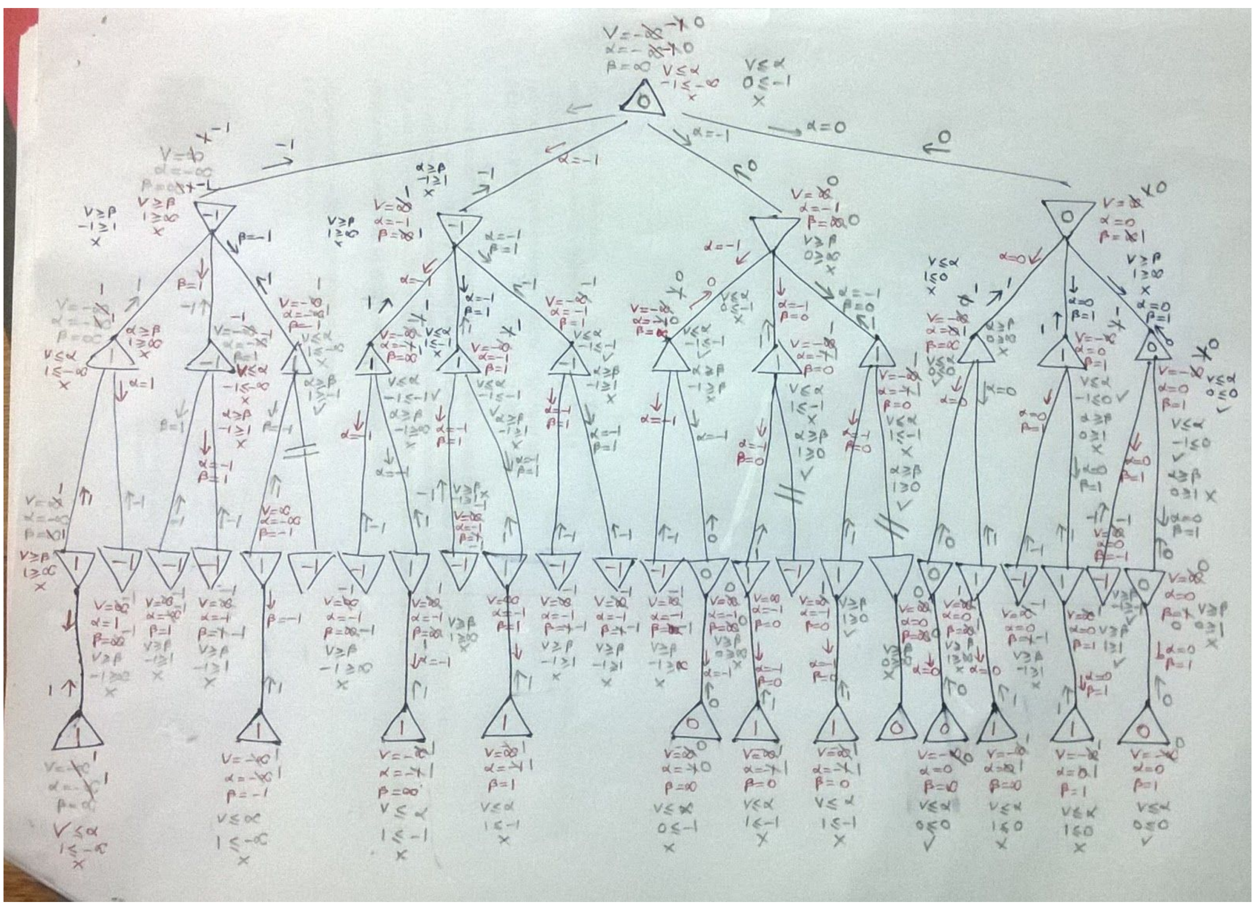
0 draw
+1 P1 win
–1 P2 win (P1 lose)

In this game P1 is X
P_1 has started the game
So MAX is P_1 ; MAX(X)

if $\alpha \geq \beta \rightarrow$ prune







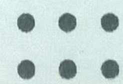
Exam Questions: pen and paper game: 2015 Exam Question 2

Question 2 (a) What is meant by *adversarial* search? **(2 marks)**

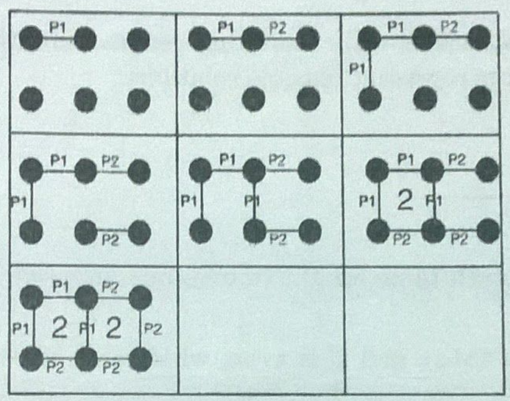
(b) Describe the **minimax** algorithm for adversarial search. **(4 marks)**

(c) How is **alpha-beta pruning** an improvement on minimax search? **(4 marks)**

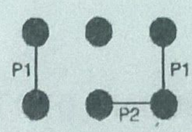
(d) Consider a two-player game, played with pen and paper, defined as follows. The game begins by drawing six dots:



Players take turns to connect adjacent dots either horizontally or vertically. If a move completes a square then the player wins the square, marks it, and takes an extra turn. The object is to win both squares. A game involves 7 moves in total. An example game, won by player 2 is as follows:

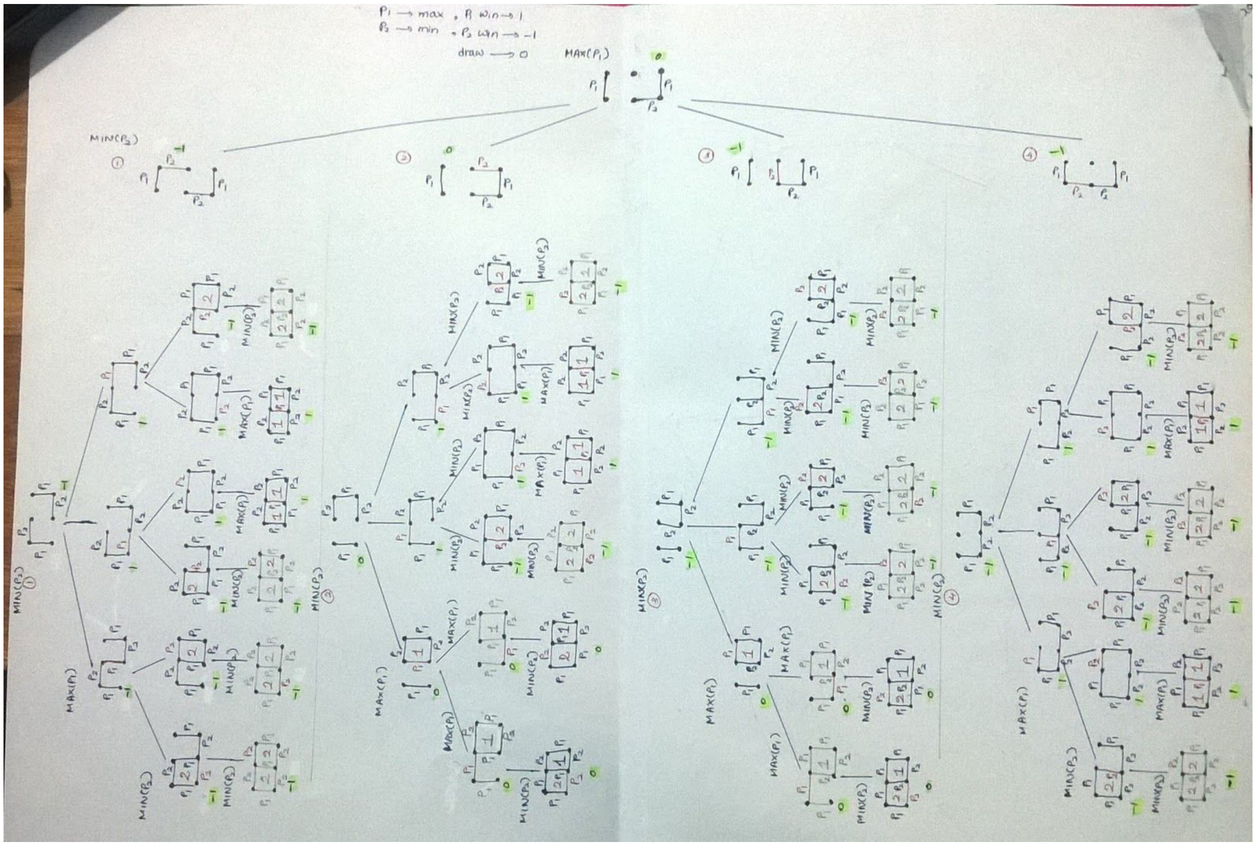


Consider the game in the following state, where it is Player 2's turn to move.

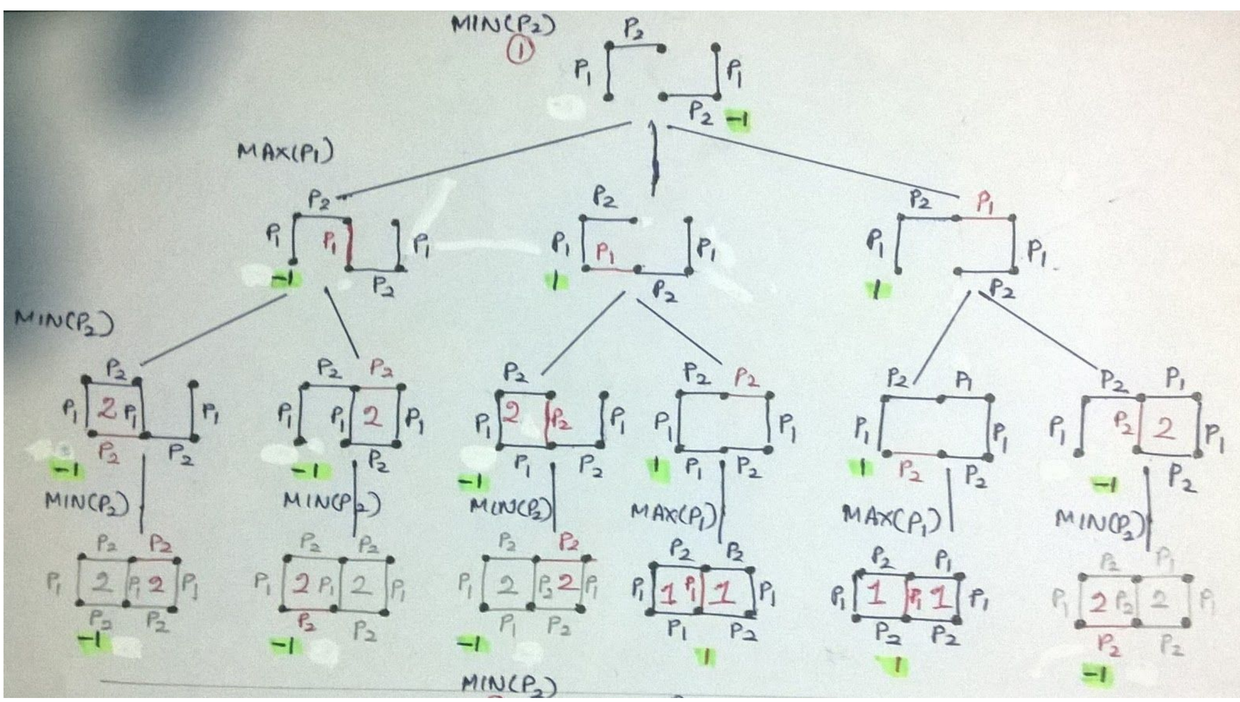


i. Draw the game tree for this game starting from this state. **(6 marks)**

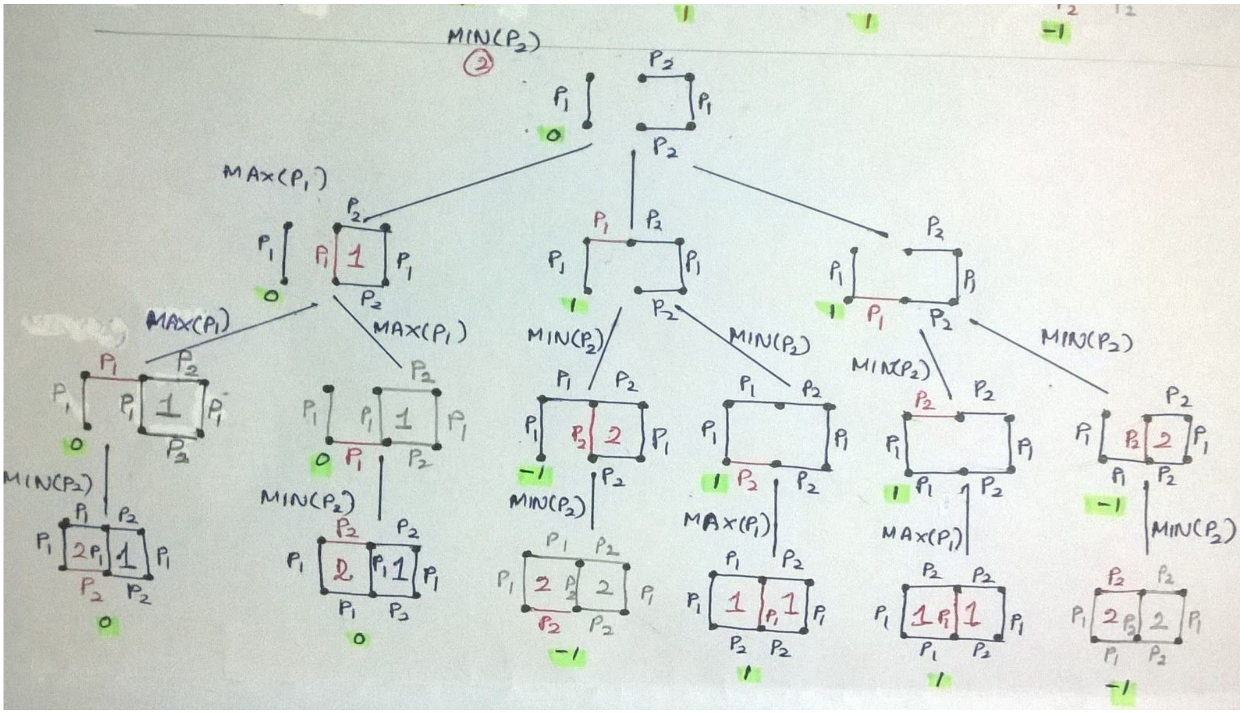
ii. Use the game tree and the minimax algorithm to show the outcome for each of the four possible moves, assuming each player plays the best move on each turn. **(4 marks)**



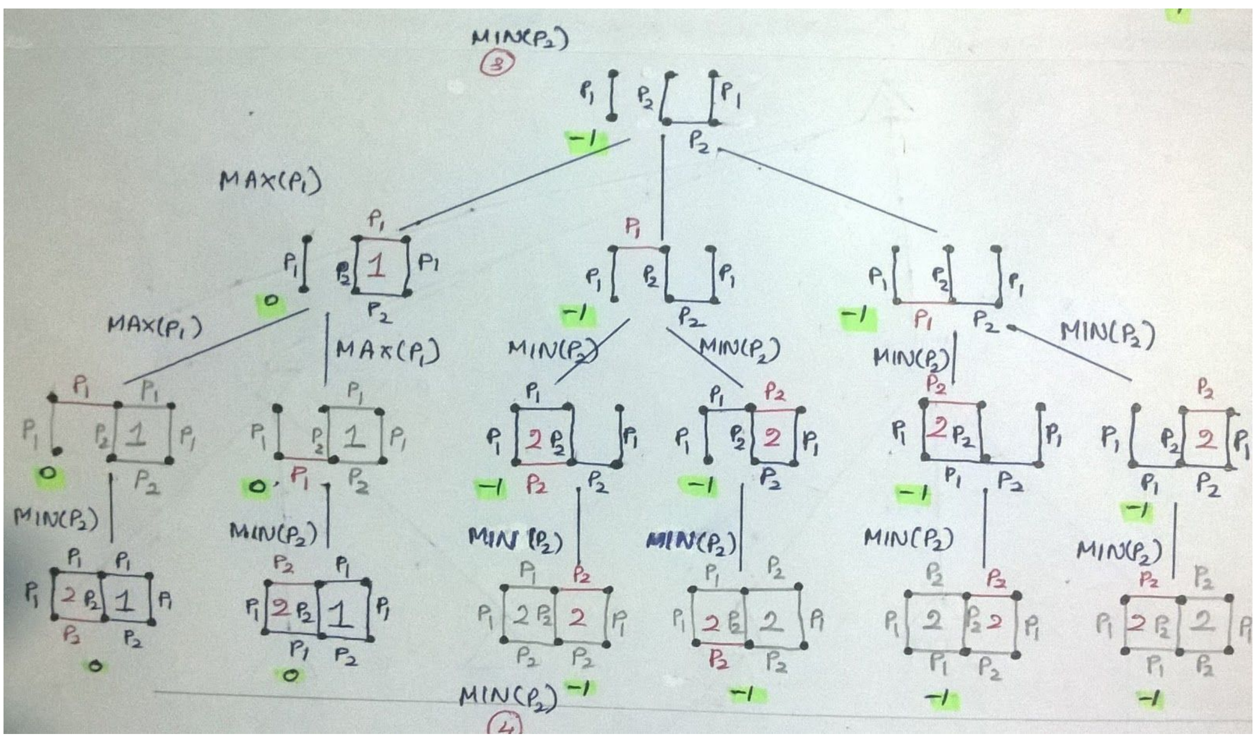
Sub tree 1



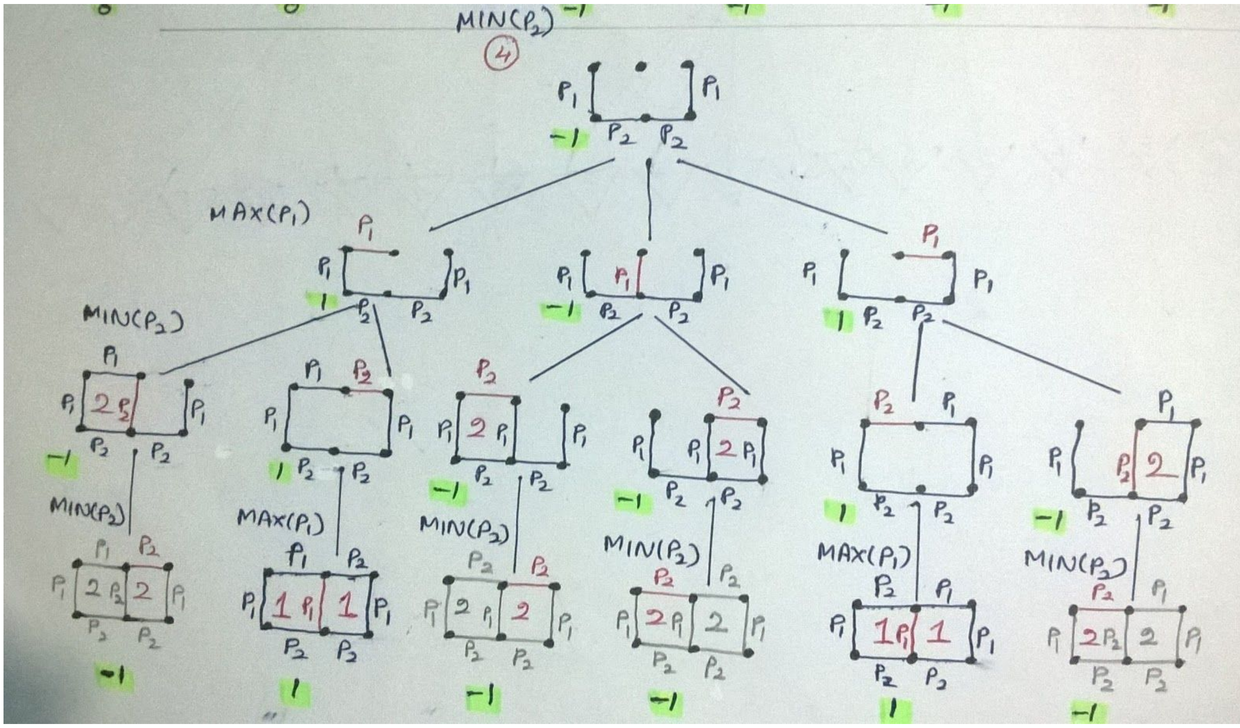
Sub tree 2



Sub tree 3



Sub tree 4



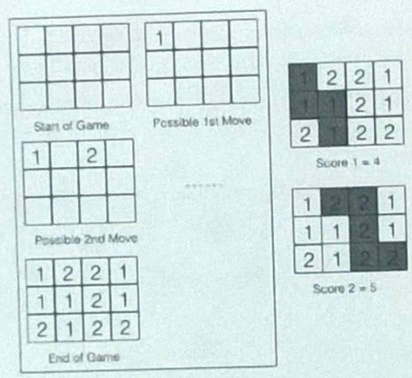
Exam Questions: connect squares : 2016 Exam Question 2

Question 2 (a) i. What is meant by *adversarial* search? **(2 marks)**

ii. Describe the **minimax** algorithm for adversarial search. **(3 marks)**

iii. How is **alpha-beta pruning** an improvement on minimax search? **(3 marks)**

(b) Consider a two-player game, played in a rectangular grid of 12 squares as shown below. On each successive turn, a player selects a square and enters his player number in it. By the end of the game, each player's score corresponds to the size of the largest connected set of squares containing his number. Two squares are connected if they share an edge. If the game ends as shown below, then Player 1's score is 4, because the largest connected set of 1's is 4. Player 2's score is 5. Since Player 2 has the largest score, Player 2 wins the game.



Consider a point in the game with 3 moves remaining and the current state of the game as shown below, with Player 2's turn to play:

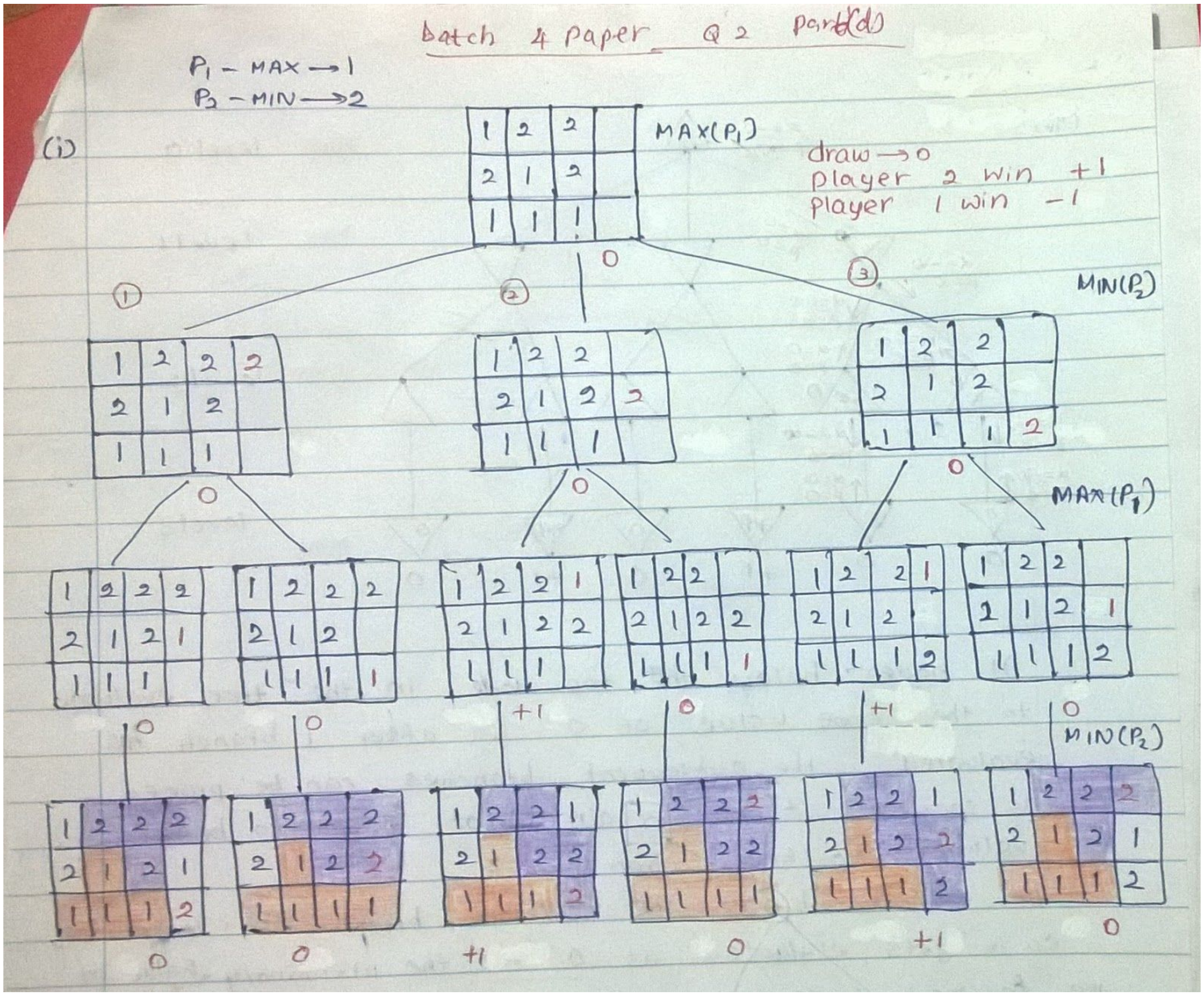
1	2	2	
2	1	2	
1	1	1	

i. Draw the minimax game tree for this game starting from this state. **(4 marks)**

ii. Is it possible for Player 2 to win the game from this point? **(2 marks)**

iii. Use the game tree and the minimax algorithm to decide which is the best move for Player 2 to make. **(4 marks)**

iv. Suggest an order in which the game tree could be expanded so that some branches would be pruned by alpha-beta pruning. **(2 marks)**



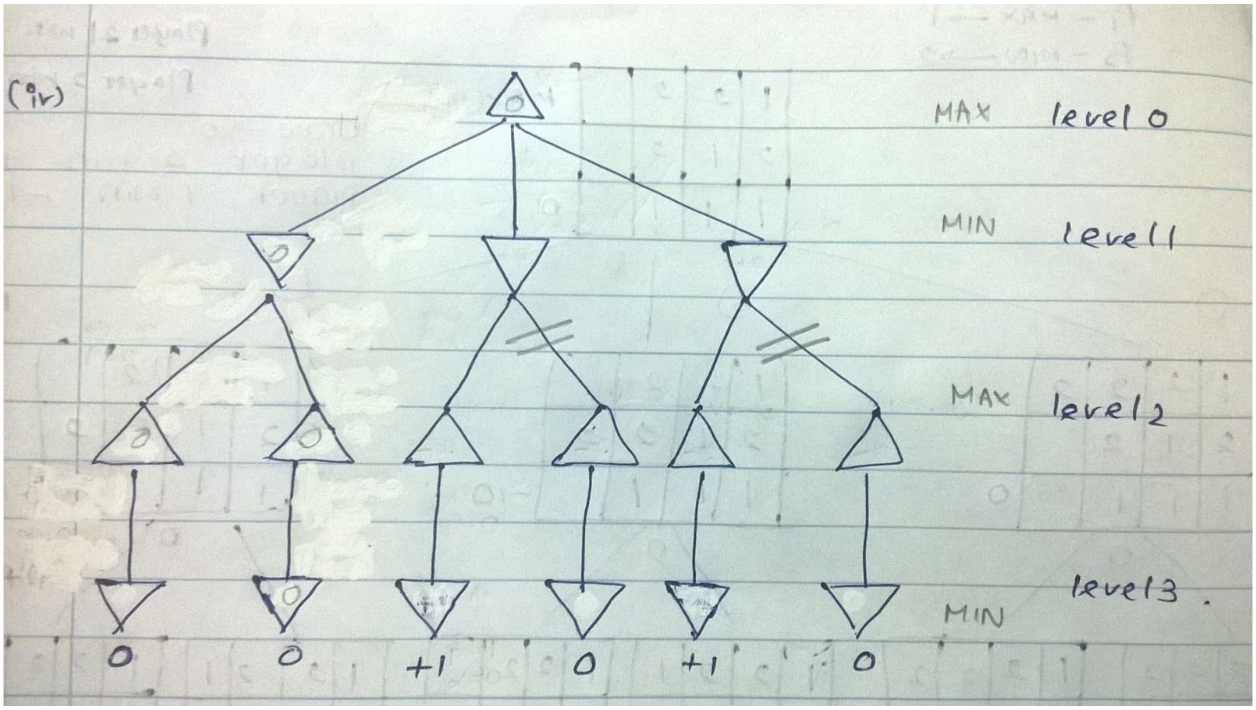
ii) yes, it is possible to player 2 to win the game.
There is a winning position , the utility value of +1 in at least one of the leaf nodes.

iii) the all three moves have utility 0, so any of the moves can be chosen for player 2 as the best move, where player 2 will not lose.

iv) Alpha beta pruning

All nodes below the top node in the tree evaluate to the same value of 0. So, after one branch has evaluated, the subsequent branches can be pruned as soon as it is certain hat there is best value is no better than 0.
All branches below the leftmost branch are traversed, so it gets evaluated as 0 and the preliminary value of the topmost node is 0. From the middle branch, going left, we get an preliminary value of 0. Since this is a MIN level, it means that its' final value can be no greater than 0, so that the middle is never better than the left. So, we can prune the right branch from the middle node on level one. Similarly, we can prune the right branch below the rightmost node

on level one. So, going in the order of the picture will allow some alpha beta pruning to be done.



2017 Exam

Question 2 (a) i. What is meant by *adversarial* search?

(2 marks)

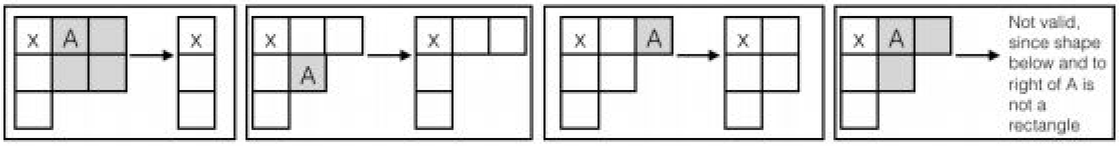
ii. Describe the **minimax** algorithm for adversarial search.

(3 marks)

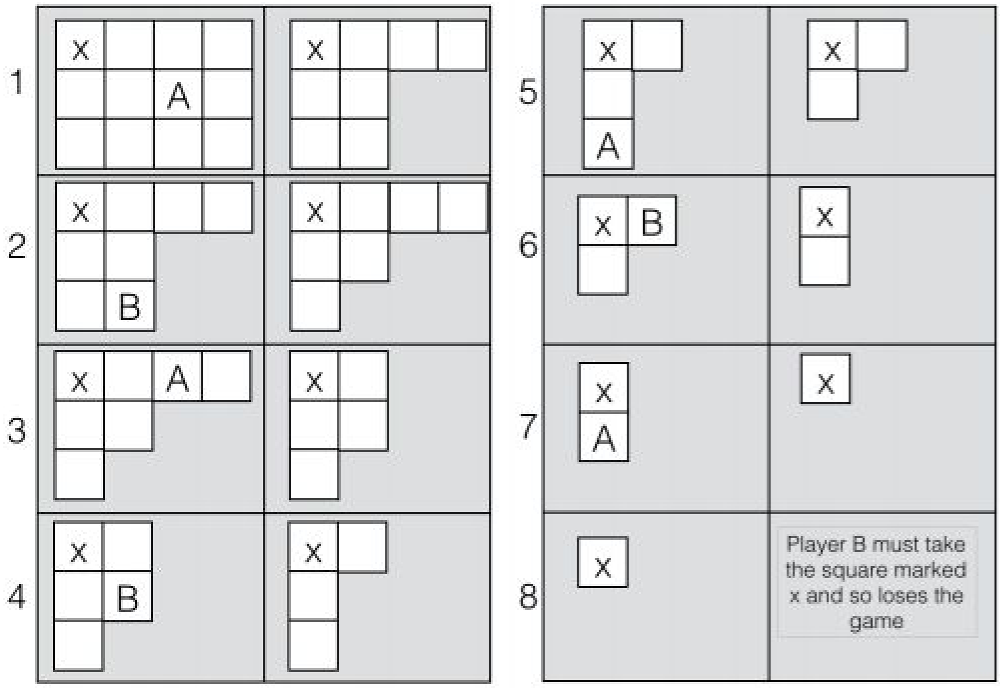
iii. How is **alpha-beta pruning** an improvement on minimax search?

(3 marks)

(b) Consider a two-player game between players A and B, played in a rectangular grid of 12 squares as shown below. On each successive turn, a player must choose a square. The rectangular block to the right and below that square is removed from the grid. The player who must finally choose the upper-left corner loses the game.
Three valid moves and one invalid move for player A are shown in the diagram below:



A full game of eight moves is shown below:



Vocabulary

Adversary - opponent - විරුද්ධකාරයා, ප්‍රතිමල්ලවයා
Outright - සම්පූර්ණයෙන්
Widen - පළල් කරනවා, විශාල කරනවා
Strategy - උපායමාර්ගය
Utility - ප්‍රයෝජනවත් කම
Alliance - සම්බන්ධය
Disrupt - කඩාකප්පල් කරනවා
Prune - කප්පාදු කරනවා