ASSIGNMENT : 3

Problem 1

1) The game tic-tac-toe, the three variants of minimax algorithm - the simple version, alpha-beta search & depth limited search will terminate in computing the best next move because the game has the finite number of moves, using the above variants of minimax will always terminate. The total number of moves is 9!.

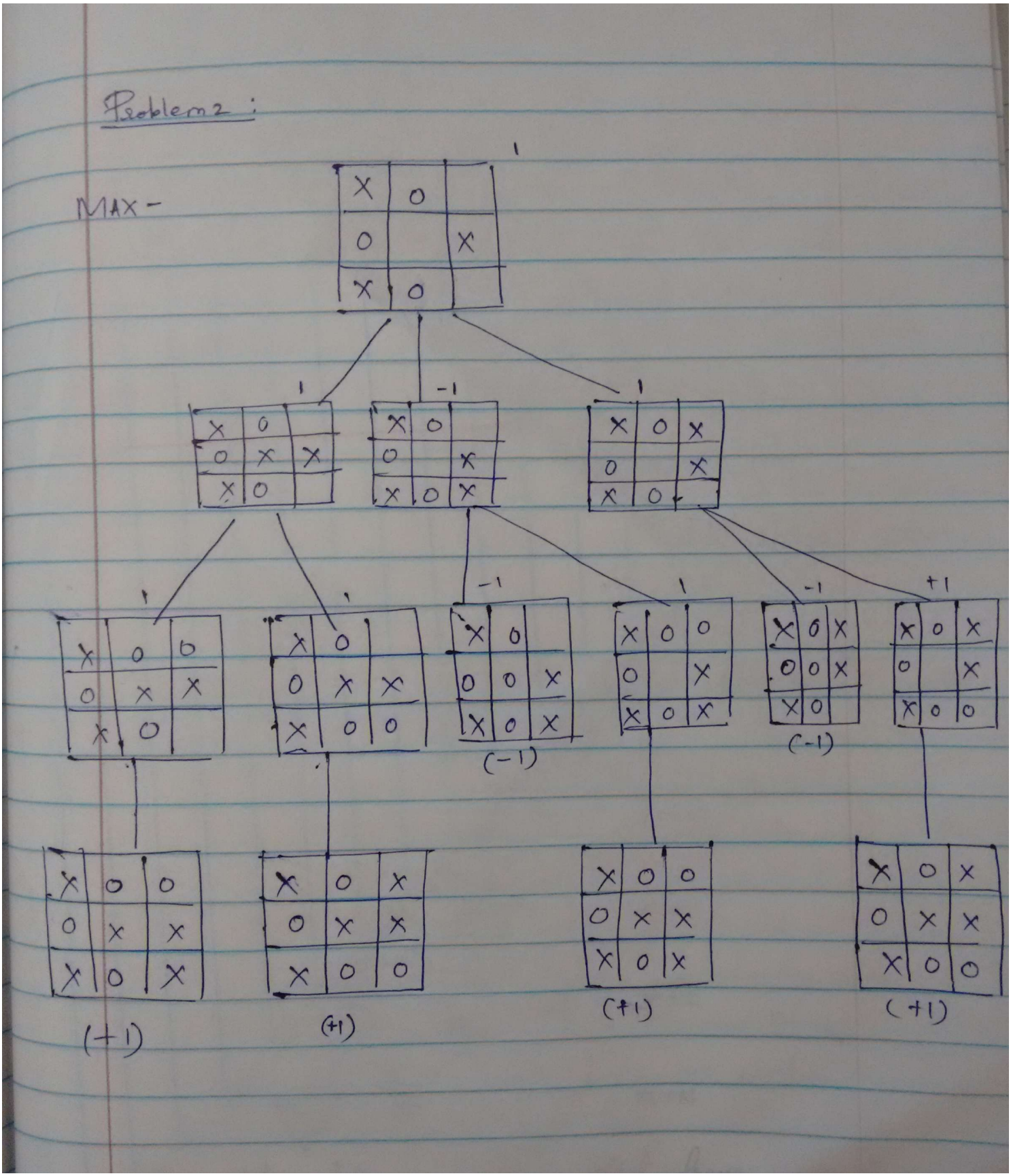
Incase of chess,- the simple version & the alpha beta search will there is a possibility that it will not terminate because the simple version has time complexity of $O(b^m) = 35^{100}$ so the solution is completely not feasible.

The Alpha beta search has complexity of $O(b^{m/2}) = 35^{50}$ which is also a huge number, hence still impossible.

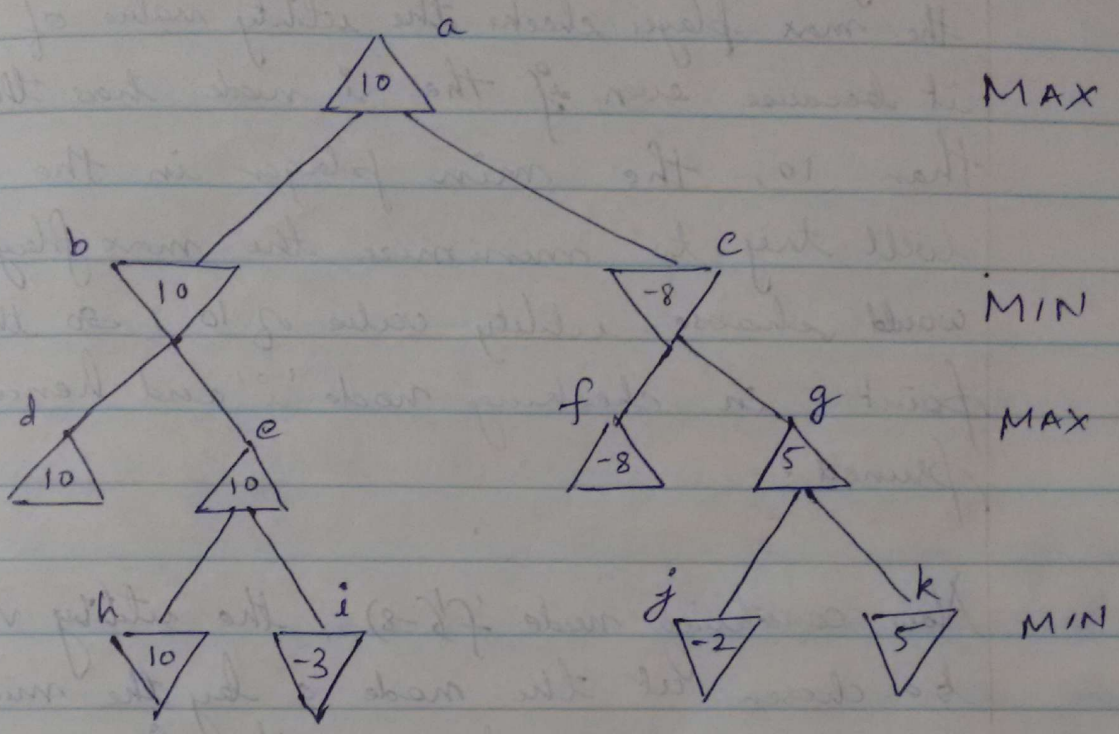
In Depth limited search the search will terminate as we will specify the maximum depth limit in making a search hence the search will terminate & will not be infinite.

Problem 2 :

MAX -

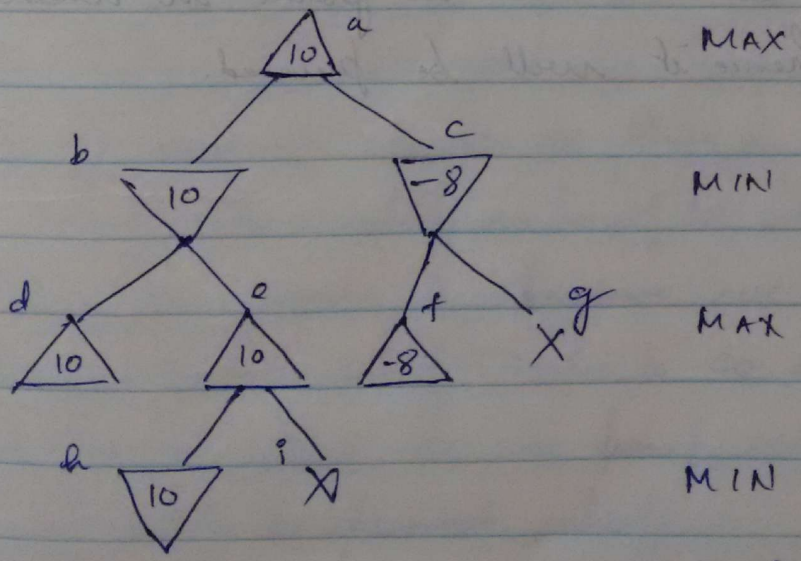


Problem 3



The above figure shows a game tree search with all the utility values.

(a)

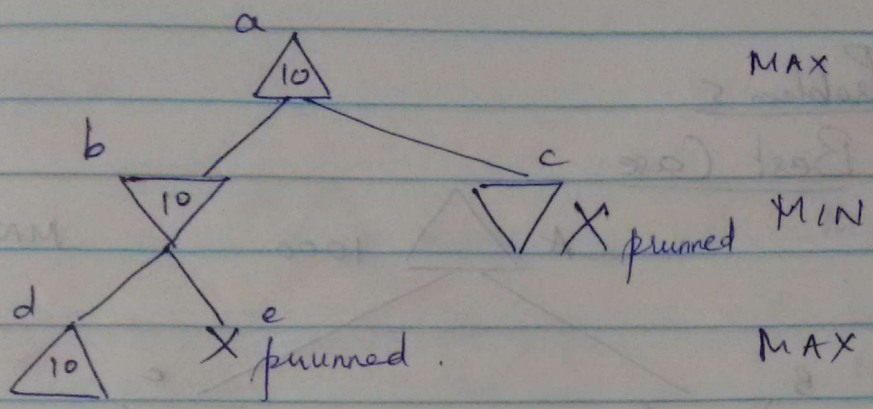


The alpha beta search will start from node 'd' which has the utility value 'd', next comes the node e with 10 as the

utility value which has 2 child nodes h (10) and i(-3) the max player checks the utility value of h, chooses it because even if the i node has the value greater than 10, the min player in the next step will try to minimize the max player utility & would choose utility value of 10. So there is no point in checking node 'i' and hence it will be pruned.

Now considering node f(-8), the utility value -8 will be chosen at the node e by the min player, the node 'g' will be pruned this is because even the node g has the utility value (5) the min player in the next step will choose the minimum utility value i.e. -8, so there is no point in checking the g node hence it will be pruned.

(b)



Given in the problem the maximum utility value is 10 and cannot be greater than 10.

With this additional knowledge the number of nodes evaluated will be decreased. The nodes that will be pruned are 'e', 'h', 'i', 'g', 'j', 'k' from the given figure. The 'b' node will have the utility value of 10 from 'd' node because, the min player chooses a minimum value & the node 'e' will never be evaluated because if the utility value is greater 10 it will the min player at the next step will choose node with utility value 10 i.e. 'd' so there is no point in exploring it. next the utility value of 10 at node B, will be chosen by node A because we already know that the maximum utility value is 10 and cannot be greater than 10 so there is no point in evaluating node c so it will be pruned entirely.

Problem 4

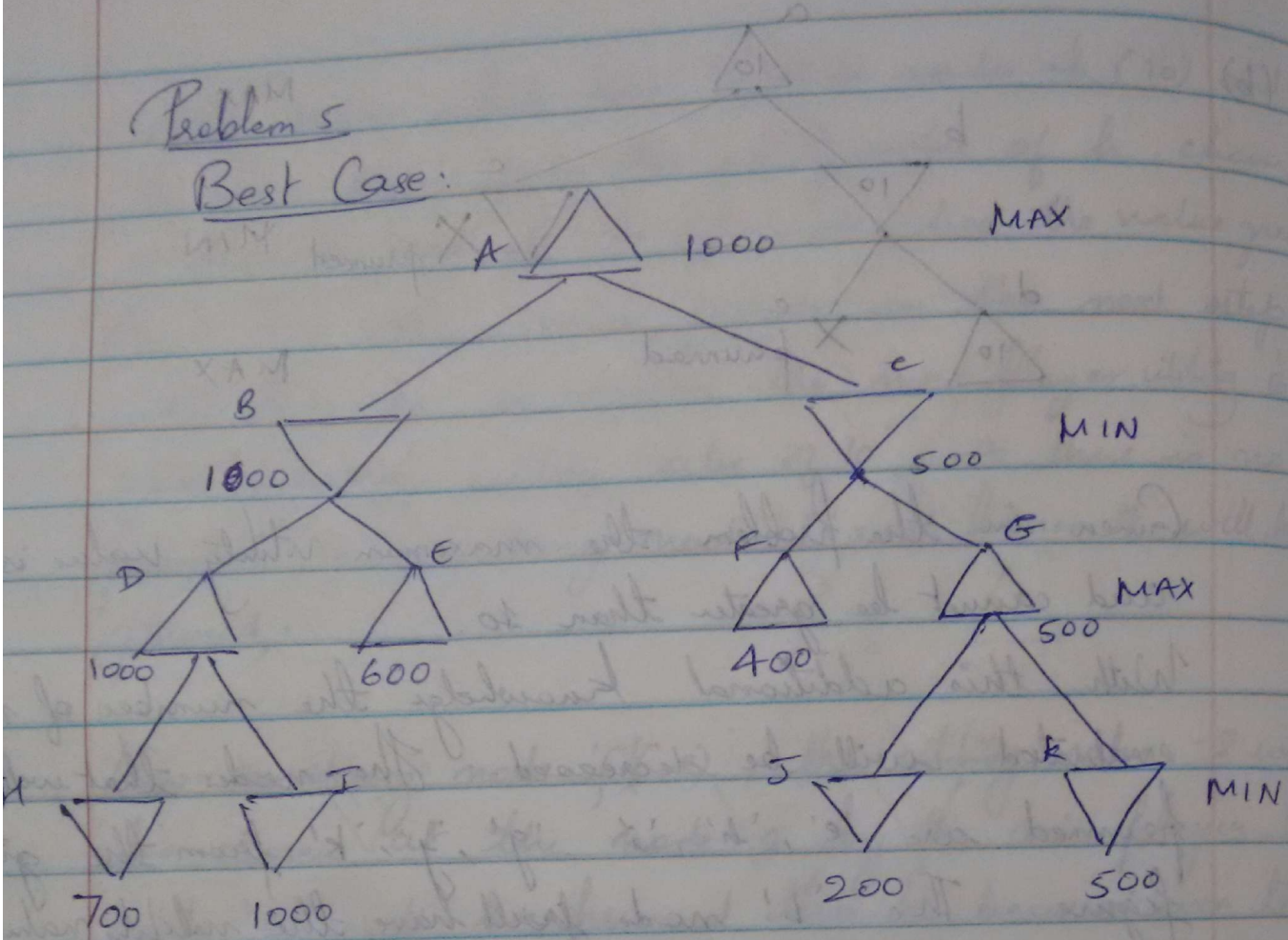
Given that 2 computer algorithms are playing against each other and algorithm A is a plain minimax search

(a) Even if the B is a plain minimax search, the algorithm A will follow the best possible strategy since the algorithm uses minimax search. The opponent B will always try to minimise the utility value of A. if we assum A is the max player. & the max player will always try to maximise its utility value. Both players will play the optimal game.

(b) No, it is not possible to replace minimax with better strategy because the minimax steely is optimal game move. So it will always try to maximise its ability to win whereas the min player would minimise the chances of opponent winning.
Since both A & B would always play optimal move, hence it is not possible to replace A's algorithm.

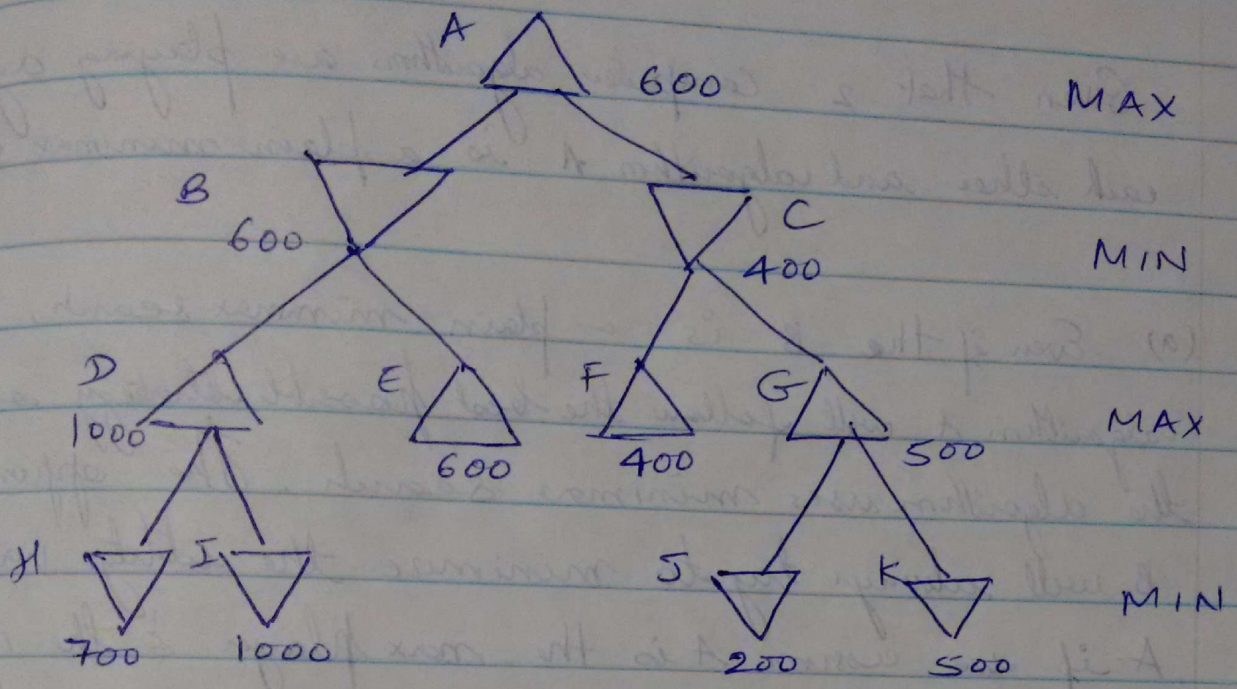
Problem 5

Best Case:



Assuming that the max player will always make the optimal move and the ~~incorrect~~ player min player will not make an optimal move.
The max player ~~is assumed~~ to make optimal move as it uses minimax algorithm.
Now, as the min player (opponent) not making an optimal move it will choose 1000 at node B instead of choosing min 600 value. Similarly at node C it will choose 500 instead of 400. Hence the max player will have the utility value of 1000 ~~are~~ in the best case scenario.

Worst Case:



Since the MAX player is using the MINIMAX algorithm, it will always make an optimal move. Whereas in the worst case the MIN player was assuming that it uses the optimal algorithm it will choose the optimal node at each of its move.

Now the min player (opponent) at node B would choose 600 as it will try to minimize the max player utility value. Similarly at node C, the min player would choose 400 the minimum utility value among F(400) & G(500).

So in the worst case the max player utility value would be 600.

Problem 6

We use a minimax algorithm to compete with the opponent since it is deterministic game of perfect information. Since the opponent is Deepgreen does not use minimax so we will use this DeepGreenMove(s) library function to get the state resulting from the opponents move.

While writing the pseudocode I have assumed that the opponent is min player.

Below is the pseudocode:

function MINIMAX-DECISION (state) returns an action
input : state, current state in game.
return the a in ACTIONS(state) maximizing DeepGreenMove(Result(a, state))

function MAX-VALUE (state) return a utility value
if TERMINAL-TEST(state) then return UTILITY(state)
$V \leftarrow -\infty$
for a, s in SUCCESSORS(state) do $V \leftarrow$ MAX(V, DeepGreenMove(s))
return V.