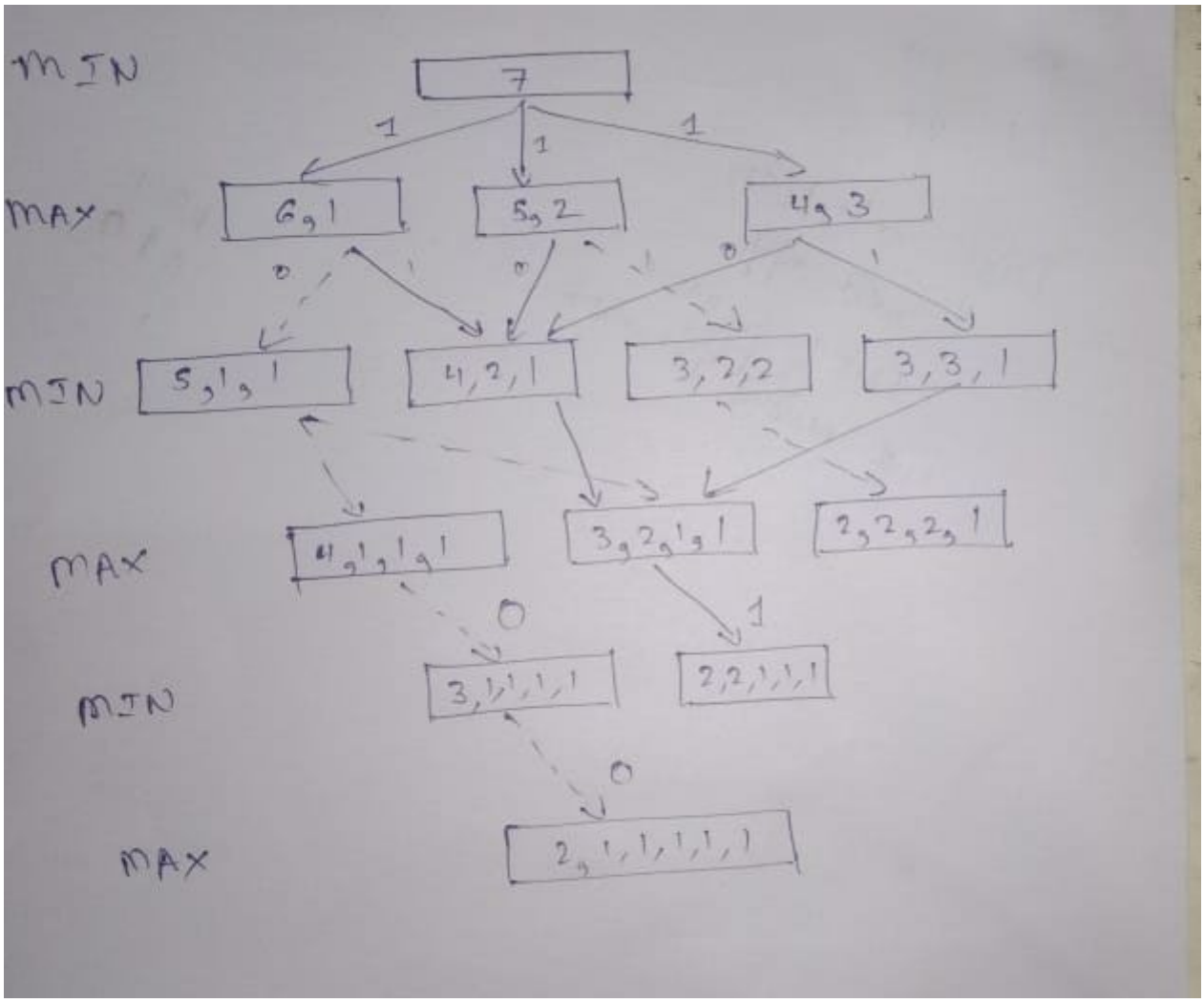
Artificial Intelligence
Assignment 2

Submitted by: Aditya Choudhary
Roll Number: 2K17/CO/024

Q1. Draw the game tree for Grundy's game: Two players have in front of them a single pile of objects say stack of 7 pennies . The first players divides the stack in to two unequal pile , second player does the same until all piles of two or one object left . The player who plays last is a winner. Work out the steps of Minimax on it.

A1: Let the first player is the 'minimum' player and the second player is the 'maximum' player.'Min' takes the first move. as the rules of minimax, if a terminal state is get by the 'minimum' player a function assigns it the value '0'.function of '1' is assigned if the same is done by the 'maximum' player.
Now, let us apply the algorithm of minimax for the given problem of Grundy's game with a stack size of 7.



The straight lines indicate the winning of the 'maximum' player (or rather forced win). Meanwhile, the dashed lines show the winning of 'minimum' player. The division of stacks are indicated by a commas and number of coins is indicated by the numbers. Numbers outside the node indicate the winning of a particular player. i.e. as said above if 'minimum' wins its 0 and if 'maximum' wins it.

Q2. Is the minimax procedure a depth first or a breadth first search procedure?

Justify your answer.

A2: Minimax is a recursive algorithm which is used to get the optimal maxima for a player letting other player to play optimally. During game play we must know what of moves are possible and then move according to it.

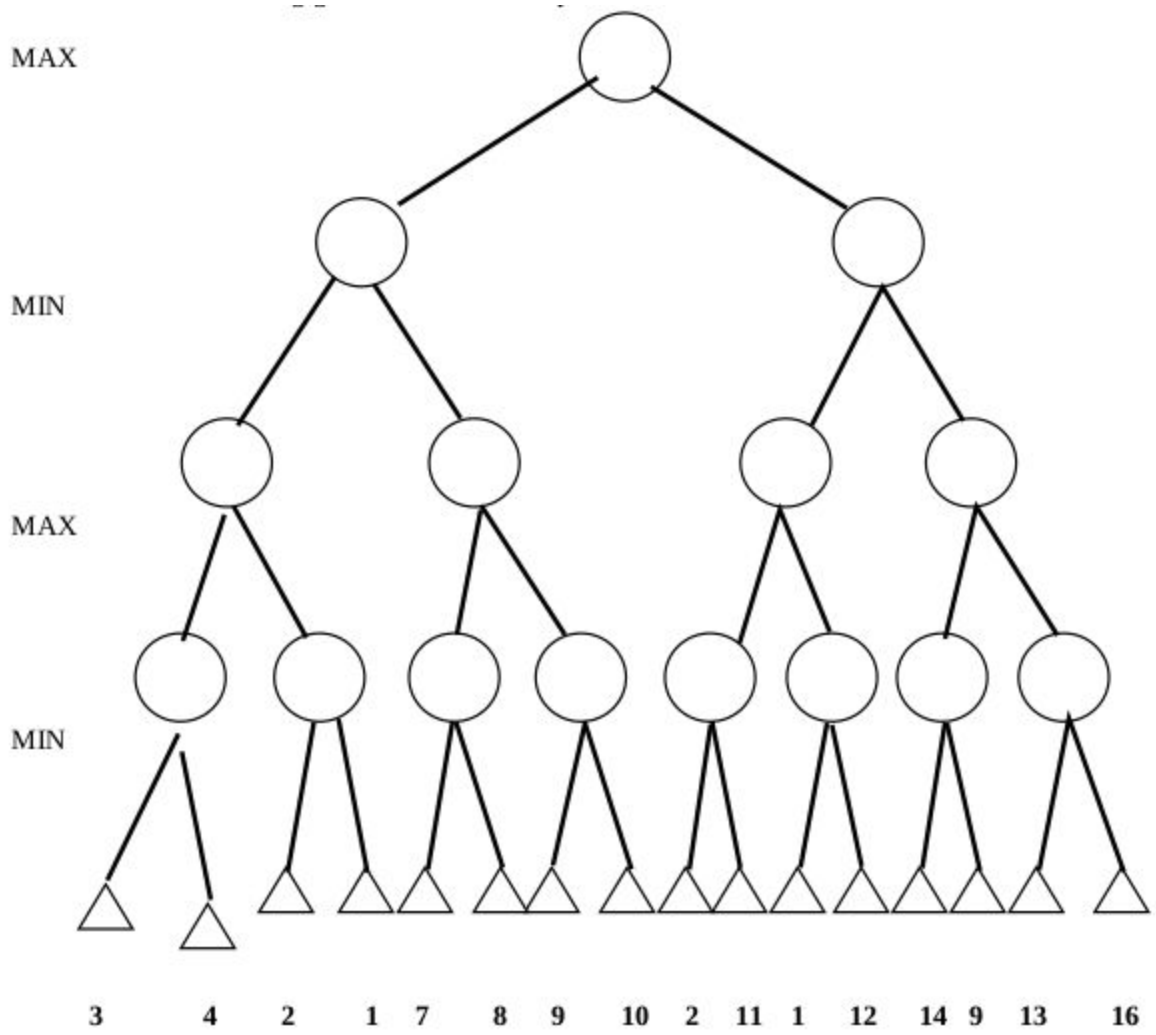
Minimax procedure ensures that if one move is taken at one point of time. Essentially it becomes a backtracking procedure. Minimax is an algorithm which traverses down to a depth and treats the nodes as if they were the terminal nodes, calling heuristic function to determine their values.

Assigning values recursively back up the tree (proves the backtracking scenario of the minimax algorithm).

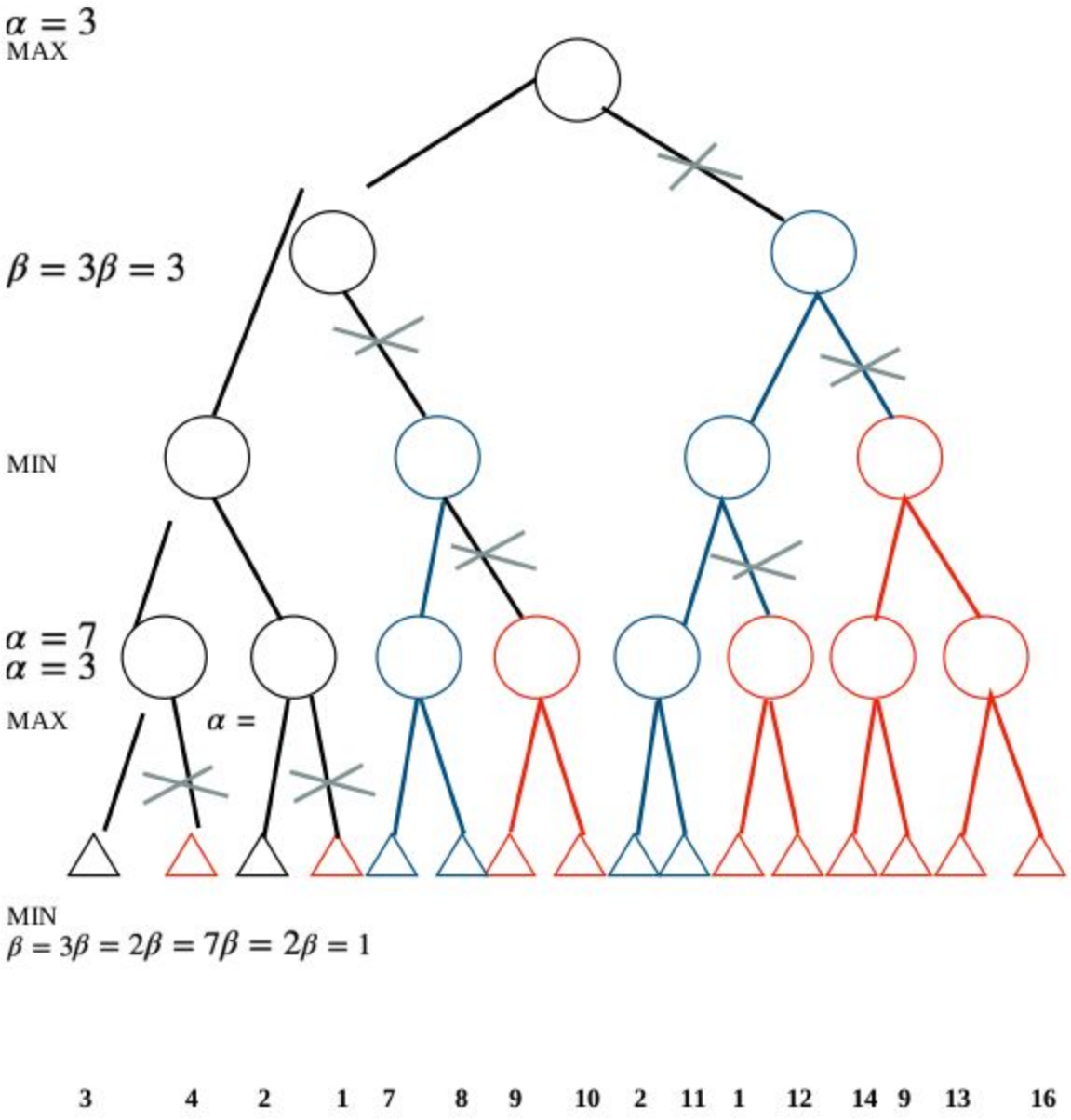
Hence, this proves that instead of searching for all the nodes at a certain level it is good to search for a node which gives the best answer and thus minimax works better with a depth search algorithm. So, Minimax can be better represented by a dfs algorithm.

Q3. Discuss the role of adding alpha-beta cutoffs in minimax search through an appropriate example.

A3: α − β Pruning is an updated form of the Minimax Algorithm. It prunes the subtree which do not follow the constraints of the minimax tree.We will use the following game tree as an example.



PRUNED GAME TREE



By introducing alpha-beta cutoffs to minimax search procedure in game playing, it avoids exploring each and every node for extraction of optimal solution. Only the required nodes are searched. Through this method unnecessary nodes are identified and not explored by pruning them.It gives the added advantage of reducing the time to extract optimal solution,I.e., time complexity is reduced to a greater extent.The time complexity of traditional minimax search procedure is O(b^d) whereas the time complexity after introducing alpha-beta cutoffs to the search procedure is O(b^d/2), which is comparatively lesser than the original search procedure.Pruning is an extended form of the Minimax Algorithm. The whole algorithm follows the exact steps as that of minimax algorithm but few more constraints are set up with it.

Constraints-

• The ‘min’ player will be minimise the score of its opponent.

• The ‘max’ player will try to maximise its own score.

• The constraint α is the current best value of α at that level/node that guarantees the maximum value for the ‘max’ player.

• The constraint β is the current best value of β at that level/node that guarantees the minimum value for the ‘min’ player.

In this search procedure, all the nodes corresponding to a specific level in the game tree is associated with either alpha cutoffs or beta- cutoffs which is done alternatively for each level with root node being initially associated with alpha-cutoff. Searching is initiated through the root node and executed in a depth first search procedure. It means that the new value of beta must be less than 3 when searching the child nodes.If another child node has the value of 4,then beta value remains unchanged.This least beta value is given to the alpha value for the upper level node.Now suppose the alpha=3(upper level node of beta=3). It means that the new value of alpha must be greater than 3. Otherwise the value remains unchanged.Now another child nodes of alpha=3 are searched for.Suppose the lower level beta node has a value of 2 when it had reached the terminal value.Now it is fixed that beta value would not be greater than 2 at all in any case.But we know that we need to find a value higher than alpha=3(upper level node).So it would be obviously be unnecessary to explore the child nodes of beta=2 as it would never give a value greater than 3.So in this way it prunes the other nodes and does not explore it at all.The other nodes are then searched in a similar manner.This method is efficient than traditional minimax search procedure.

Q4. Consider the following sentences:

(i) John likes all kinds of food.

(ii) Apples are food.

(iii) Chicken is food.

(iv) Anything anyone eats and isn't killed by is food.

(v) Bill eats peanuts and is still alive.

(vi) Sue eats everything Bill eats.

(a) Translate these sentences into formulas in predicate logic.

A4(a):

I. $\forall x$: food(x) $\rightarrow$ likes(john, x)

II. food(apple)

III. food(chicken)

IV. $\forall x$: ($\exists y$: eats(y, x) $\wedge$ $\neg$killedby(y, x)) $\rightarrow$ food(x)

V. eats(Bill, Peanuts) $\wedge$ $\neg$killed(bill)

VI. $\forall x$: eats(Bill, x) $\rightarrow$ eats(Sue, x)

(b) Prove that John likes peanuts using backward chaining.

A4(b):

First negate the conclusion in order to start the process of backward chaining.

$\neg$likes(John, peanuts)

Using statement I from (a)

peanuts/x1

$\neg$food(peanuts)

Using statement IV from (a)

Peanuts/x4

$\neg$eats(y4, peanuts) $\vee$ killedby(y4, peanuts)

Using statement V from (a)

Bill/y4

¬eats(Bill, Peanuts)

But eats(Bill, Peanuts)

CONTRADICTION!

Our initial assumption must be false. i.e John likes peanuts.

(c) Convert the formulas of part (a) into clausal form.

A4(c):

I. ¬food(x1) ∨ likes(John, x1)

II. food(apple)

III. food(chicken)

IV. ¬eats(y4, x4) ∨ killedby(y4, x4) ∨ food(x4)

V. eats(Bill, Peanuts) ∧ ¬killed(bill)

VI. ¬eats(Bill, x6) ∨ eats(Sue, x6)

(d) Prove that John likes peanuts using resolution.

A4(d):

First negate the conclusion and then proceed accordingly.

1. ¬likes(John, peanuts)

2. ¬food(peanuts)

Using ¬food(x1) ∨ likes(John, x1) and ¬likes(John, peanuts)

3. ¬eats(y4, peanuts) ∨ killedby(y4, peanuts) Using ¬eats(y4, x4) ∨ killedby(y4, x4) ∨

food(x4) and ¬food(peanuts)

4. killedby(Bill, peanuts) Using eats(Bill, Peanuts) ∧ ¬killed(bill) and

¬eats(y4, peanuts) ∨ killedby(y4, peanuts)

5. CONTRADICTION! Using killedby(Bill, peanuts) and eats(Bill, Peanuts) ∧

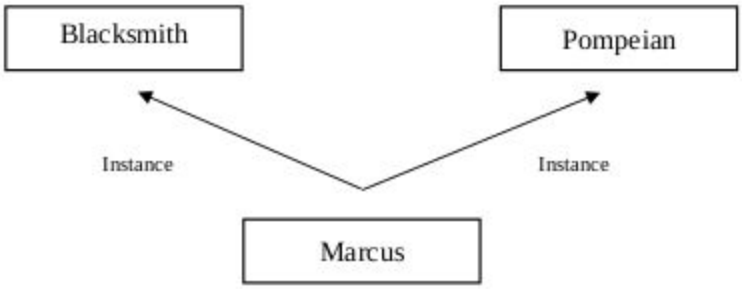
¬killed(bill).

Our initial assumption must be false. i.e John likes peanuts.

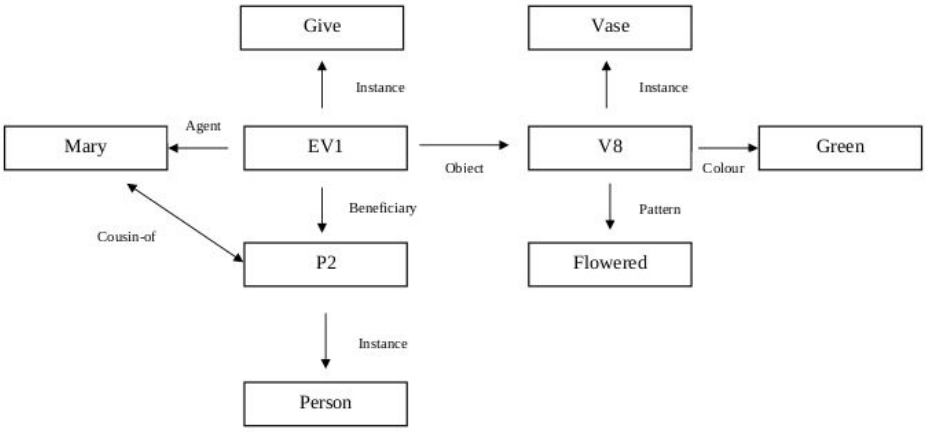
Q5. Construct the semantic net representations for the following:

(a) Pompeian (Marcus), Blacksmith (Marcus)

A5(a):



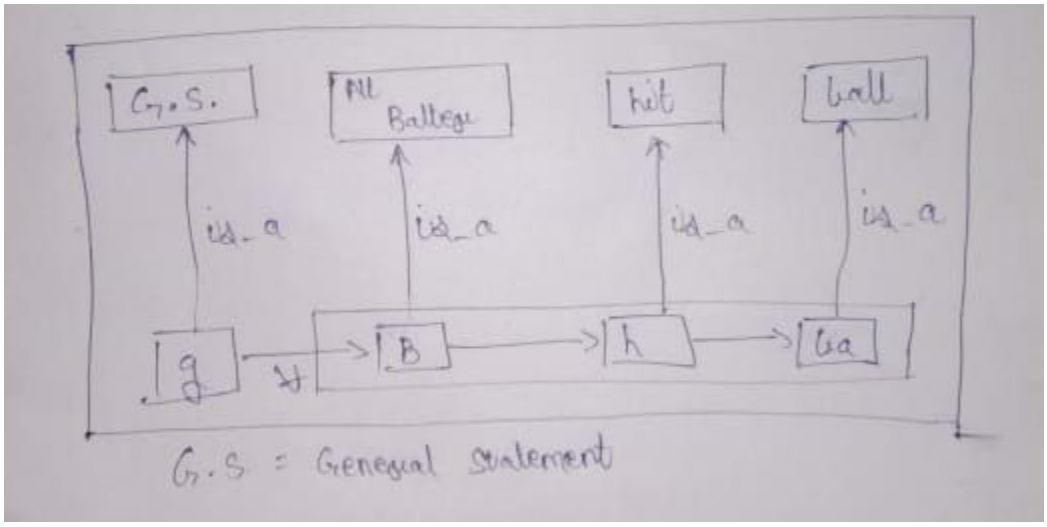
(b) Mary gave the green flowered vase to her favorite cousin.



Q6. (i) Construct partitioned semantic net representation for the following:

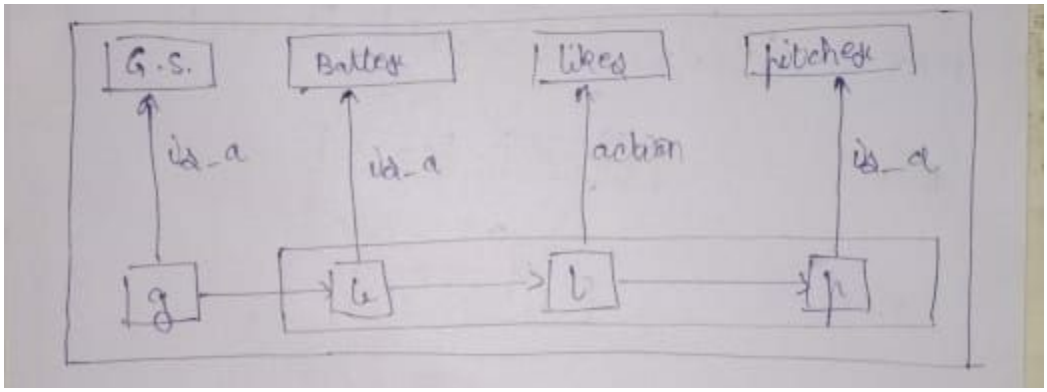
(a) Every batter hit a ball.

->



(b) All the batters like the pitcher.

->



(ii) Show a conceptual dependency representation of the sentence:

John begged Mary for a pencil.

How does this representation make it possible to answer the question:

Did John talk to Mary?

->