Solutions to FIT5047 Tutorial on Problem Solving as Search: Irrevocable and Adversarial Search

Exercise 1: Irrevocable Search Algorithms

(a) What happens in Simulated Annealing if the temperature remains at ∞ at all times? What happens if the temperature starts at 0?
SOLUTION:

- If the temperature remains at ∞, the algorithm always accepts a new state regardless of whether it is better or worse. Also, the algorithm will go into an infinite loop.
  The ∞ temperature pertains to worse states. If we reach that stage in the algorithm, a worse state will be accepted with probability 1. However, Best-So-Far will keep only the best state.
- If the temperature starts at 0, the algorithm will perform only one iteration, and return the best of the initial state or new state. Best-So-Far will keep only the best state.

(b) A genetic algorithm is to be used to evolve a binary string of length n containing only 1s. The initial population is a randomly generated set of binary strings of length n.

i. Give a suitable fitness function for this problem.
SOLUTION:
Sum of the genes

ii. Calculate the probability of selecting each of the following strings according to your function.
SOLUTION:

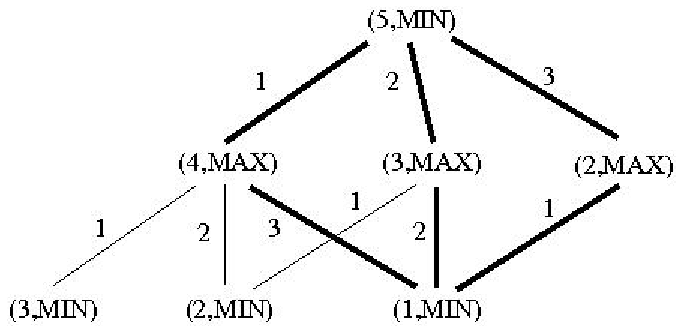
```
00110001     3/{3+5+7} = 0.2
01011101     5/{3+5+7} = 0.333
11101111     7/{3+5+7} = 0.467
```

Exercise 2: Game Graphs

The game *nim* is played as follows: Two players alternate in removing one, two or three coins from a stack initially containing 5 coins. The player who picks up the last coin loses. Show by drawing the game graph, that the player who has the second move can always win. Can you think of a simple characterization of the winning strategy?

SOLUTION:
The characterization of the general winning strategy is to leave the opponent with $1, 5, 9, 13, \ldots, 4n+1, \ldots$ coins. You can figure this out from the solution below by trying 6,7,8,9 coins (if MAX has 6, 7 or 8 coins, s/he can make sure MIN is left with 5 coins and lose, so the next winning number is 9, and so on).

The following diagram contains the game graph, where the winning paths are highlighted. Note that for **every** move that MIN makes, MAX must have **one** winning move.



Exercise 3: α-β Search

Consider a hypothetical game with branching factor 2. It is MAX's turn to play. He is able to evaluate a position 4 steps in advance. The following is a list of the values of the positions at the bottom of the game tree, should they ever need to be evaluated:

node #	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
e(n)	3	9	1	8	0	13	6	20	2	4	7	5	11	14	8	13

(a) Draw a game tree, as evaluated by MAX using the α-β procedure, so that only the expanded nodes appear in your diagram. Indicate clearly the backed-up value of each node.

(b) How many α cut-offs have been performed?

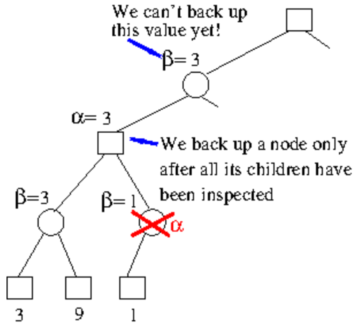
(c) How many β cut-offs have been performed?

(d) What is the best move for MAX and what is its backed-up value?

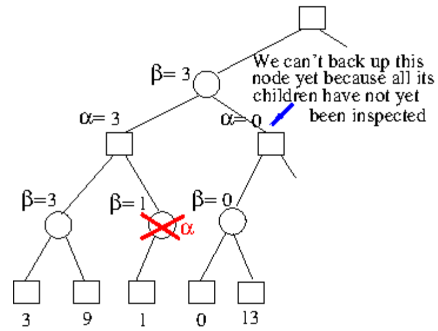
SOLUTION:
Let us first revise the concepts:

- The α-value of a MAX-node is set to the current largest **final** backed-up value of its successors. That is, you can not back up a node until you have finished looking at its children.
- The β-value of a MIN-node is set to the current smallest **final** backed-up value of its successors.

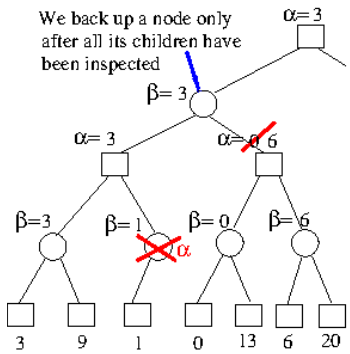
(a) The game tree is as follows.



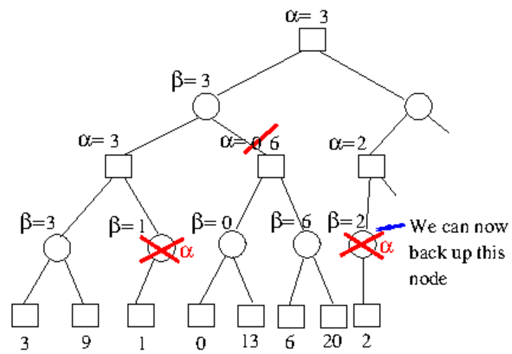
(a)



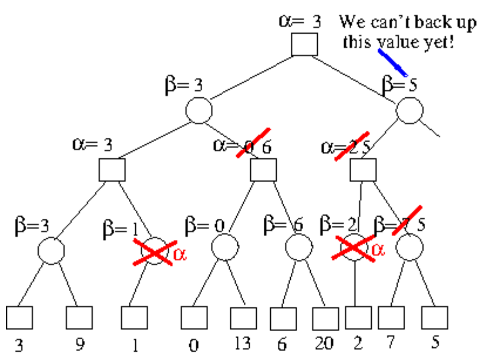
(b)



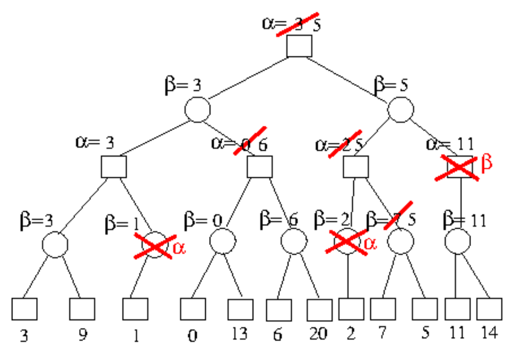
(c)



(d)



(e)



(f)

- α cut-off – search is discontinued below a MIN-node whose β value is **less than or equal to** the α value of **any** of its MAX-node ancestors. This accounts for the α cut-off **(a)** at the MIN-node with value 1, since its β-value is less than the value of its MAX-node parent, i.e., 3. This also accounts for the α cut-off **(d)** at the MIN-node with value 2, since its β-value is less than the value of the root node (which is a MAX node ancestor of this MIN node).
- β cut-off – search is discontinued below a MAX-node whose α value is **greater than or equal to** the β value of **any** of its MIN-node ancestors. This accounts for the β cut-off **(f)** at the MAX-node with value 11, due to its parent which has value 5.

Note the changes in backed-up values **(c)** because MAX prefers 6 to 0, **(e)** because MAX prefers 5 to 2, and **(f)** because the root MAX node prefers 5 to 3.

(b) There are 2 α cut-offs.

(c) There is 1 β cut-off.

(d) The best move is **to the right** with backed up value 5.