A logo for college computing

Description automatically generated

**Assessment Cover Page**

|  |  |
| --- | --- |
| *Student Full Name* | Santhosh Shanmugasundaram |
| *Student Number* | Sba24100 |
| *Module Title* | AI Concepts to Implementation |
| *Assessment Title* | CA2 |
| *Assessment Due Date* | 5th January 2025 |
| *Date of Submission* | 5th January 2025 |

**Declaration**

By submitting this assessment, I confirm that I have read the CCT policy on academic misconduct and understand the implications of submitting work that is not my own or does not appropriately reference material taken from a third party or other source.

I declare it to be my own work and that all material from third parties has been appropriately referenced.

I further confirm that this work has not previously been submitted for assessment by myself or someone else in CCT College Dublin or any other higher education institution.

CA1 – AI Concepts to Implementation

Contents

[1. Question1: Min-Max Algorithm with Alpha Beta Pruning. 1](#_Toc187005633)

[1.1 Task1a: Determine Min-Max Value 1](#_Toc187005634)

[1.2 Task1b: Determine Pruned Nodes using Alpha Beta Pruning Technique 3](#_Toc187005635)

[1.3 Task1c: Determine Alpha Beta value 4](#_Toc187005636)

[1.4 Task1d: Python program 5](#_Toc187005637)

[2 Question2: Constraint Satisfaction Problem 5](#_Toc187005638)

[3 Question3: Conquest 6](#_Toc187005639)

[3.1 Objective: Be the first player to reach opponents home (square box in the board) to conquest and win. 6](#_Toc187005640)

[3.2 Variables/Game Setup: 6](#_Toc187005641)

[3.3 Domains/Action: 6](#_Toc187005642)

[3.4 Constraints/Rules: 6](#_Toc187005643)

[3.5 Justifications: 6](#_Toc187005644)

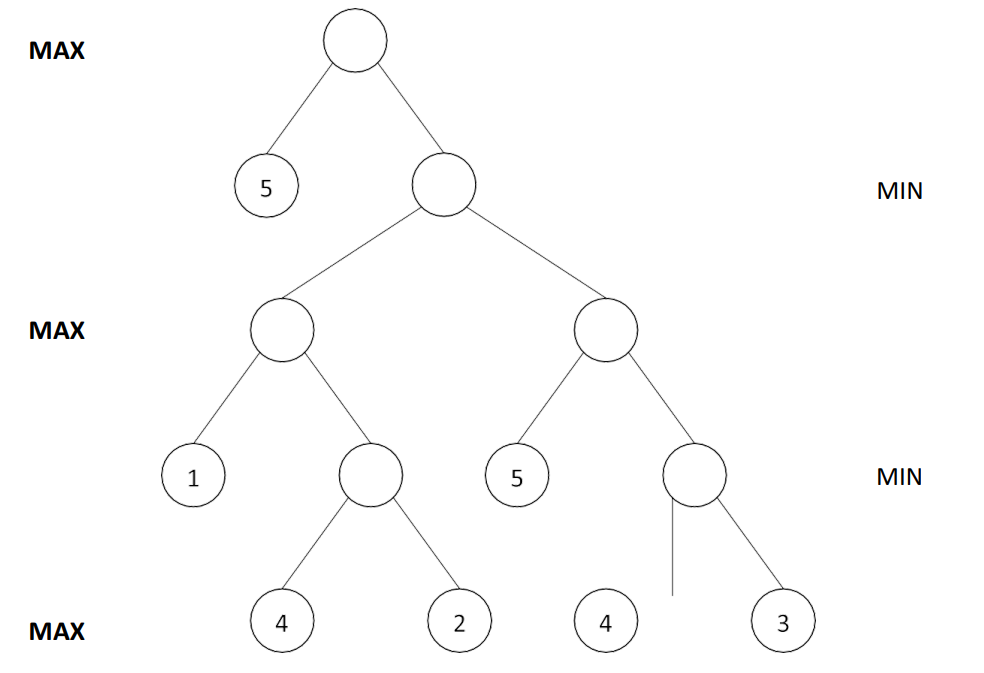
[4 References 7](#_Toc187005645)

[5 GitHub Link 8](#_Toc187005646)

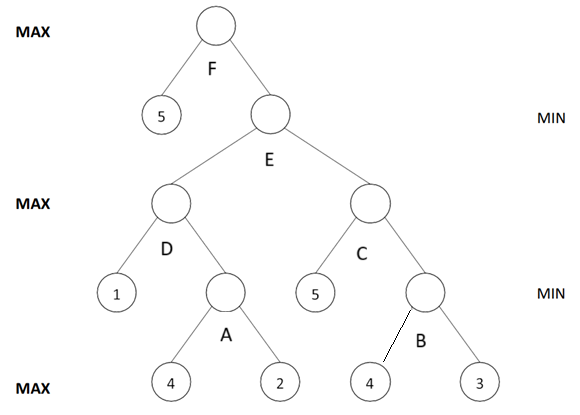
# Question1: Min-Max Algorithm with Alpha Beta Pruning.

## Task1a: Determine Min-Max Value

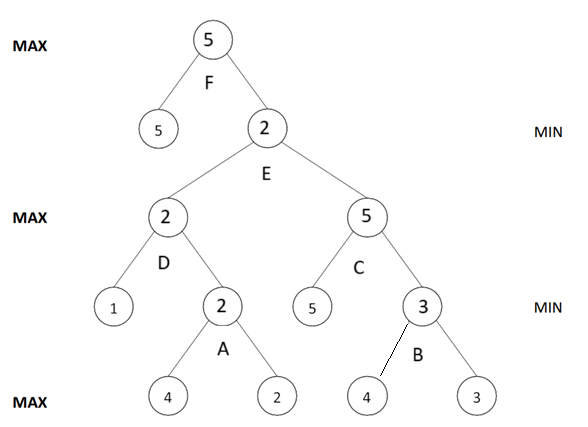
Task is to implement the min-max algorithm for the below tree and find the values at each node.

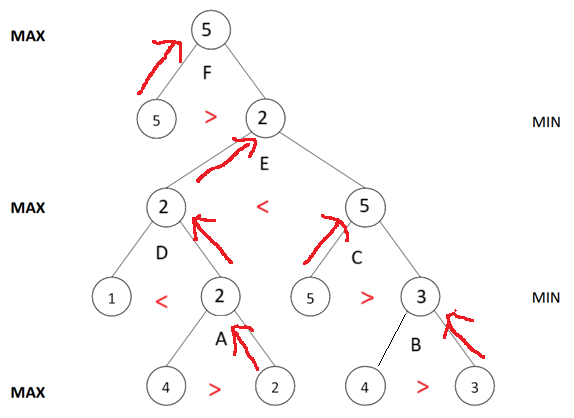


The Min-Max algorithm involves several key steps, executed recursively until the optimal move is determined. This algorithm applies Depth First Search (DFS); therefore, we must go till the terminal node. Each step involved in execution of min-max algorithm for the above tree is listed below and to start with, each node has been named from A to F starting from leaf node at bottom to top.



1. Start from leaf node, which is at the bottom, let’s try to compare the leaf values corresponding to node A.
2. Since node A and B corresponds to min node, we try to fetch the lowest of value from its associated leaf value.
3. Therefore, A is assigned as 2 (lowest of 4 & 2) and B is assigned as 3 (lowest of 4 and 3).
4. Next nodes are D and C which corresponds to Max node; therefore, we try to fetch the highest value from its associated child node/leaf node.
5. Node D is assigned as 2 (highest of 1 & 2) and C get’s 5 (highest of 5 & 3).
6. Node E corresponds to Min node; therefore, we take the lowest of D and C which results in value 2 (lowest of 2 & 5).
7. Now we arrive to the topmost/root node of the tree which contains a leaf node with value 5 and child node E whose value is 2. Taking the max value of 5 and 2, we assign 5 to F.





## Task1b: Determine Pruned Nodes using Alpha Beta Pruning Technique

Alpha-beta pruning is a technique used to improve the efficiency of the minimax algorithm by eliminating the evaluation of certain nodes that has no significance to the result.

**Alpha (α)**:   
Represents the best value the maximizing player can guarantee so far. The initial value of alpha is -Infinity (-∞).

**Beta**: Represents the best value the minimizing player can guarantee so far. The initial value of Beta is +Infinity (+∞)

Condition for Pruning

* The minimizing player finds a value that is lower than or equal to alpha (β ≤ α) then prune the branch.
* The maximizing player can prune a branch when they find a value higher than or equal to beta (α ≥ β)

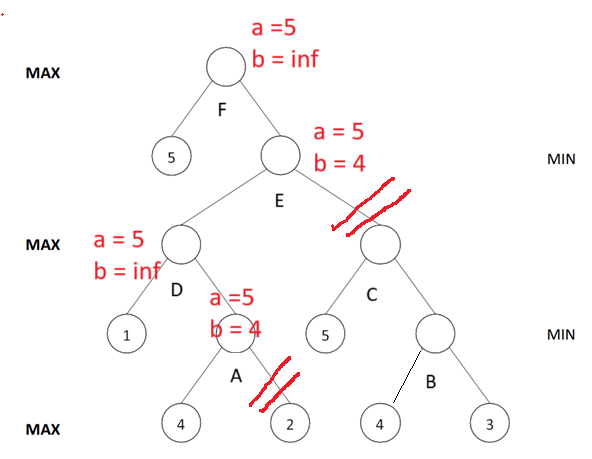
Steps:

1. Start from top node F where alpha is set to -∞ and beta to ∞.
2. Check the child node of F and set value as alpha to 5.
3. Move to node E to D and get value of 1.
4. Move to node A and get value of value beta as 4. Since beta value of 4 is less than alpha value of 5, prune the remaining child of A.
5. Move again to node E where pruning condition meets, therefore prune remaining child of E.

Solution:

For the provided tree, following nodes are pruned,

Child 2 of node A  
Right side of node E, which is C & B

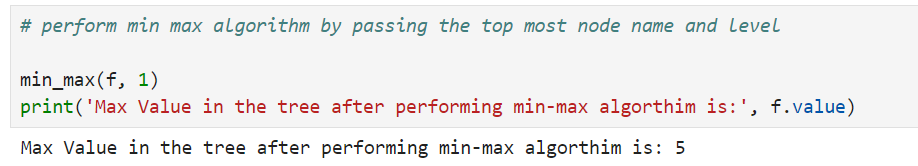


## Task1c: Determine Alpha Beta value

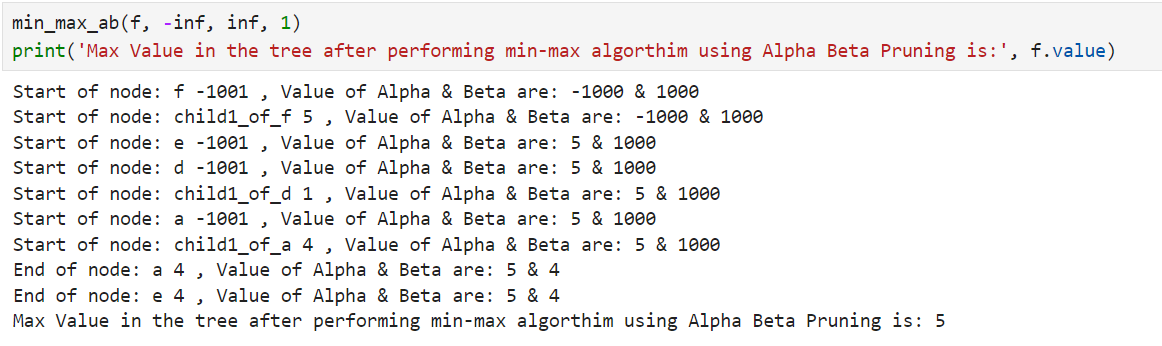
Final value of Alpha and Beta at each node is as below,

1. Node F: α = 5 and β = ∞.
2. Node E: α = 5 and β = 4.
3. Node D: α = 5 and β = ∞.
4. Node A: α = 5 and β = 4.

## Task1d: Python program



The python program result below, shows the list of nodes that were activated when using Alpha beta pruning. It does not contain nodes that are pruned.



# Question2: Constraint Satisfaction Problem

## Task2a: Representation of the Problem

**Representation1: Medical Service as Variable**

In this representation, we set medical service as a variable, meaning there will be 4 categories of variables namely – ‘Routine patient Check-up’, ‘Blood and other tests’, ‘Surgeries’ and ‘Routine Check-up’.

The domain of each variable consists of two namely – ‘Days’ and ‘Hours’.

**Representation2: Days as Variable**

In this representation, we set days as variable. Therefore, we will have 5 category variables namely – ‘Monday’, ‘Tuesday’, ‘Wednesday’, ‘Thursday’ and ‘Friday’.

The domain of each variable consists of two namely – ‘Services’ and ‘Hours’.

Considering the above information, we can say that **Representation 1** allows for simpler and straightforward implementation of constraints specific to service and the reason as below,

* 1. In representation 1, we see that the total hours of ‘Routine Check-Up’ service on both Wednesday and Friday are 7 hours.
  2. In representation 2, we see that there are two different hours of services on Wednesday.  
     Wednesday 🡪 Blood and Other Tests 🡪 4 hours  
     Wednesday 🡪 Routine check-up 🡪 7 hours

Two different hours on Wednesday adds up complexity to representation 2.

## Task2b: Total number of possible solutions

The total number of possible solutions depends on the minimum period week that we take. Considering that the minimum period week is 2 to meet equal workload for both doctors, we could have up to 280 different possible solutions.

## Task2c: Formulations

**Total Variables:**  
Considering that we have 4 different types of service that occur 6 times in a week and over 2-week time, we get it for 12 times.

Therefore, we end up creating 12 variables.

# Question3: Conquest

## Objective: Be the first player to reach opponents home (square box in the board) to conquest and win.

## Variables/Game Setup:

1. Game board with 8 x 8 white and black square grid (Chess board can be used).
2. First player will receive 4 white pawn and a white king.   
   Second player will receive 4 black pawn and a black king.
3. Each player to chooses a coloured box (colour to match the piece colour) in the top row as their starting position where king need to be placed. In the immediate row below, 4 pawns must be grouped in the boxes matching to the piece colour.

## Domains/Action:

1. **Move**: Starting with white piece, move it to an adjacent empty square in the same row or forward row.
2. **Block**: Strategically position each piece to block the opponent’s movement toward the home square.
3. **Find**: Study the opponent’s position and find the shortest path to opponent’s home square.

## Constraints/Rules:

1. White pieces need to be moved first, followed by black and this repeats equally.
2. All the pieces need to be moved forward or in the same row. Backward movement not allowed.
3. Pieces cannot move to a square that is occupied by another piece.
4. Game stops when a king piece reaches the opponents starting row.

## Justifications:

1. **Intelligence**:

The game demands that players engage in strategic thought and careful planning to move their pieces effectively while obstructing their opponent's advancement.   
Participants must assess the existing board situation, predict their opponent's actions, and develop a comprehensive strategy aimed at fulfilling their goals.

1. **Artificial Intelligence**:  
   The game can incorporate artificial intelligence algorithms to develop computer-controlled adversaries that demonstrate intelligent behaviour.  
   Techniques such as the A\* algorithm may be utilized to create AI agents that are proficient in making strategic choices and efficiently navigating the grid.
2. **Agent:**In the framework of the game, each player's piece can be viewed as an agent with individual task for pawn piece and king piece.   
   Players function as agents who manage their pieces and make decisions informed by the prevailing conditions of the game and their strategic goals.
3. **Rationality:**Players strive to make logical choices that enhance their likelihood of winning the game.   
   In this scenario, rationality means choosing actions that not only improve one's own standing but also hinder the opponent's advancement.   
   The game incentivizes those who demonstrate sound decision-making by effectively moving their piece to the opponent's starting point.
4. **Logical Reasoning:**Players are required to use logical reasoning to evaluate the implications of their actions and foresee possible results.   
   This reasoning is applied to examine the grid configuration, determine the best routes, and anticipate the potential moves of their opponents.

# References

(Saxena)

Saxena, Abhimanyu. “Alpha Beta Pruning in Artificial Intelligence.” *Applied AI Blog*, 25 Sept. 2024, www.appliedaicourse.com/blog/alpha-beta-pruning-in-artificial-intelligence/. Accessed 9 Nov. 2024.

Available at:   
<https://www.appliedaicourse.com/blog/alpha-beta-pruning-in-artificial-intelligence/>

# GitHub Link

<https://github.com/santhosh-sba24100/CA1---AI-Concepts-to-Implementation>