Game Trees

# Introduction to game trees

So to start out this topic we will be playing a game. In this game 2 players are in an intense stand off, plucking petals of a flower. You each take in turns, plucking either 1, 2 or 3 petals of each go. Whoever plucks the last flower off the flower loses.

## The Game

So lets play!

Game 1: the number of petals on the flower is 6. I will take the first go. (remove 1).

Let them take their go.

Remove the amount of petals it takes to get to 1.

Then you win.

Game 2: the number of petals on the flower is 13. Let them go first.

Now remove the number of petals to get to 9.

Their go.

Get to 5 petals now.

Their go.

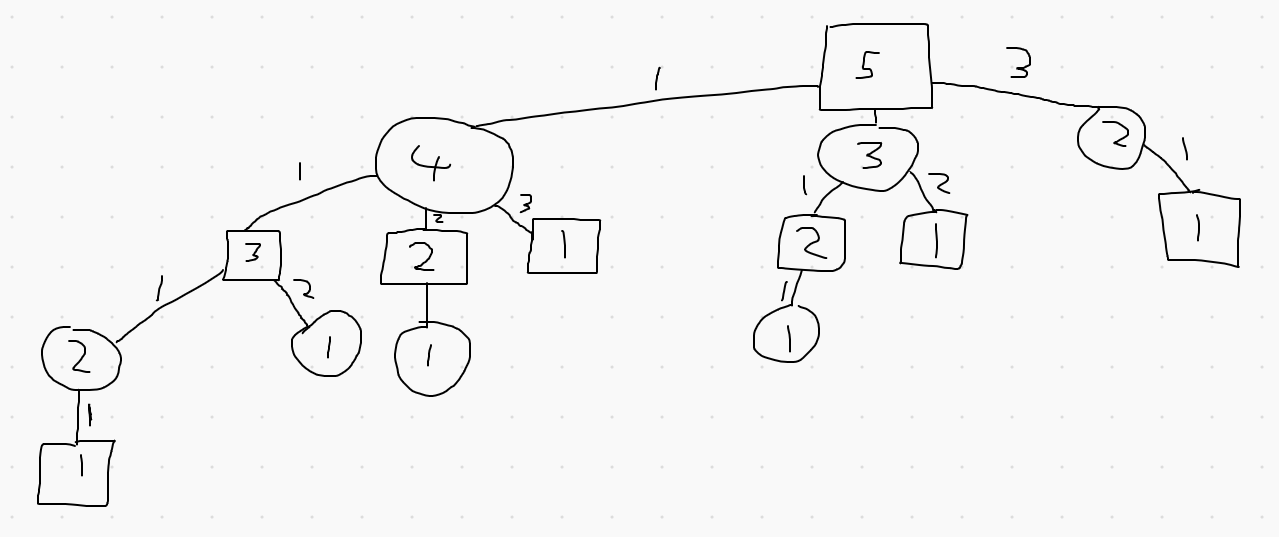
Get to 1 petal now.

Then you win.

What I have shown here is the power of game trees by always being able to win as I got to decide what number of petals to start. Using game trees we will be able to know whether a game is winnable if both players play **optimally**.

## Game trees

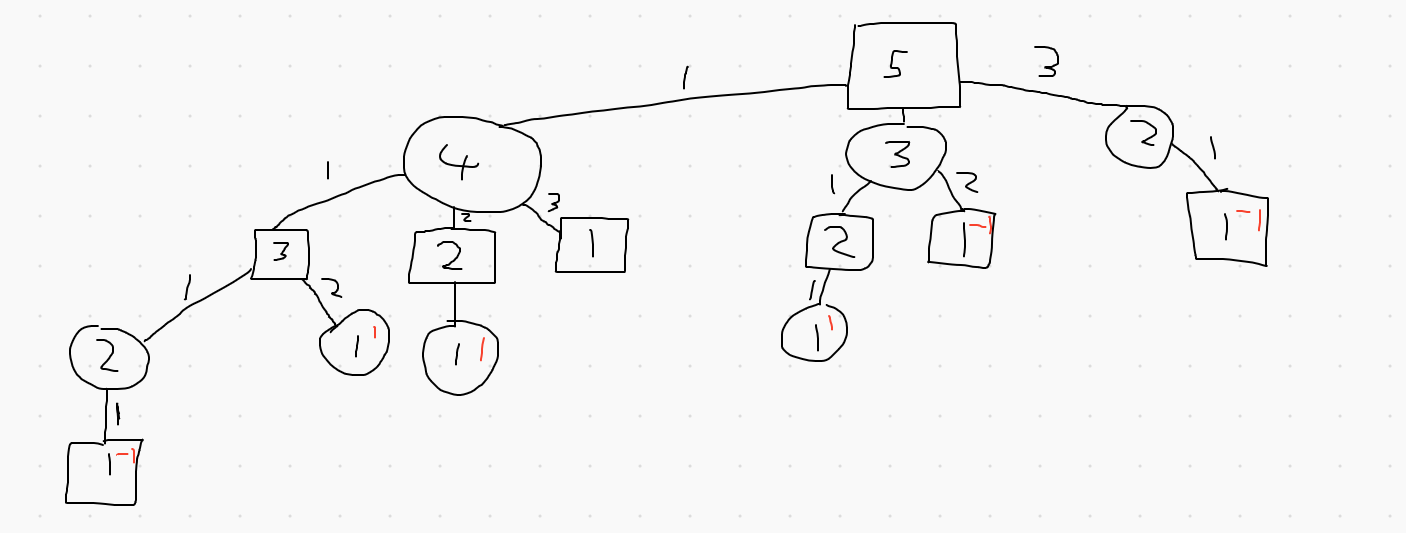
Game trees are a method of representing decisions being made in a game on a tree. For the game above here is the tree of when the petals = 5.



The squares represent player1’s go and circles represent player2’s go. So any leaf node is a loosing state. The connections show the move they player took, so if the connection is labelled 1 it means the player in the parent node removed 1 petal. The number in the centre is the number of petals.

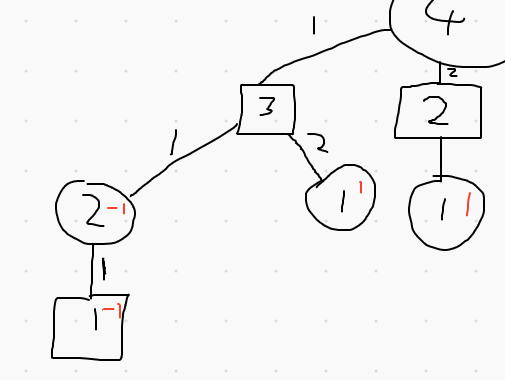
We will be looking at this game from the “perspective” of player 1, so we will be seeing if player 1 can win. For each node in the graph we want to calculate it , where is the value of the number of petals. This value ranges from 1 to -1, if 1 it means if player 1 is in this state they will win, same goes for -1 but it means they will lose.

So we start at the leaf nodes and work our way up, as if you are at the leaf node you know who will win and who will lose ( is represented in red):

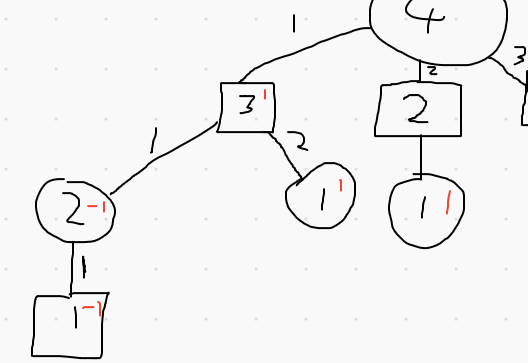


(I missed a leaf node in the child of 4)

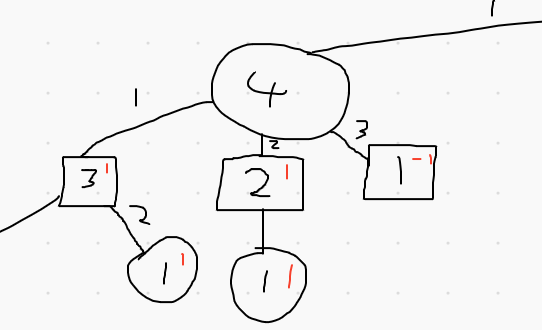
Now we will work our way up the tree. We assume here that both players will play optimally. So if the current node is a square (player 1) we will always choose the max of the child nodes. This is the opposite for a circle node (player 2) they will always choose the min of the child nodes. If there is only 1 child node then the parent node just takes on the children’s . So starting on the far left:



The value is inherited from the child. Then we reach our first decision, it is a square node, so we take the max ox the child nodes, so :

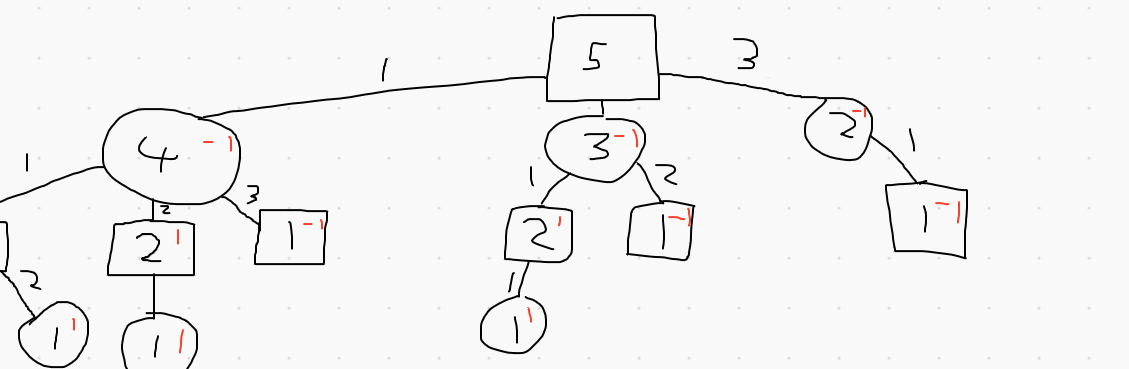


Skip forward a bit:



This is a circle node, so the minimum will be taken.

Skip more until you get to the root node:



So we’ve reached the root node, all options are -1, this means that if player 2 plays optimally then player 1 cannot win. So .