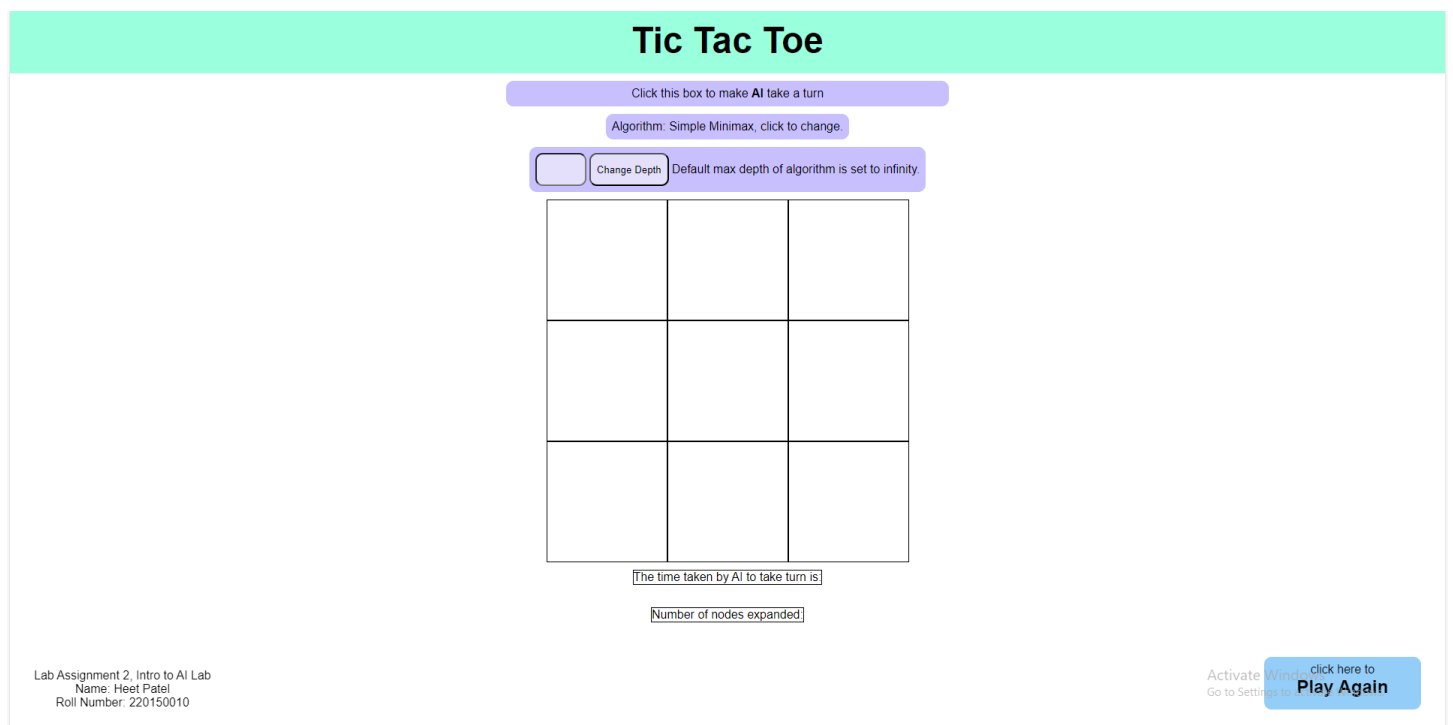


Tic Tac Toe

using MiniMax with and without alpha/beta pruning

I Designed the Tic Tac Toe game in Vanilla JS using the Minimax search algorithm, with and without alpha pruning.



Here's the link to the hosted game:

<https://heet434.github.io/tic-tac-toe-minimax/>

There are features to make AI take turn at any point of time, change algorithm to with alpha-beta pruning and without alpha-beta pruning and to change the depth function according to the user.

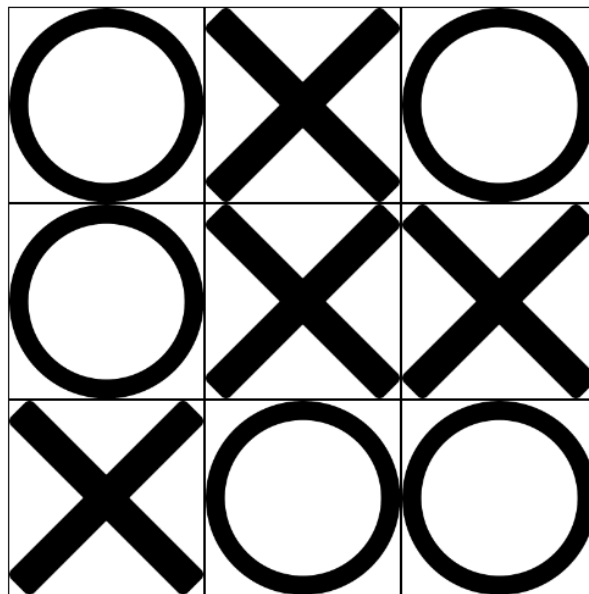
Number of nodes expanded per algorithm and time taken per algorithm are also displayed.

A graph generated towards the left compares the time taken per algorithm on each turn.

Comparing the two algorithms:

Here is the graph for a complete game (tie game) shown below:

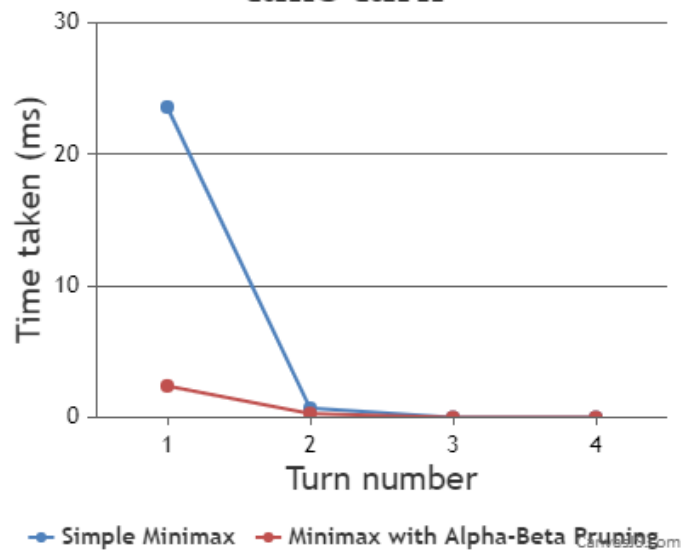
It's a tie!



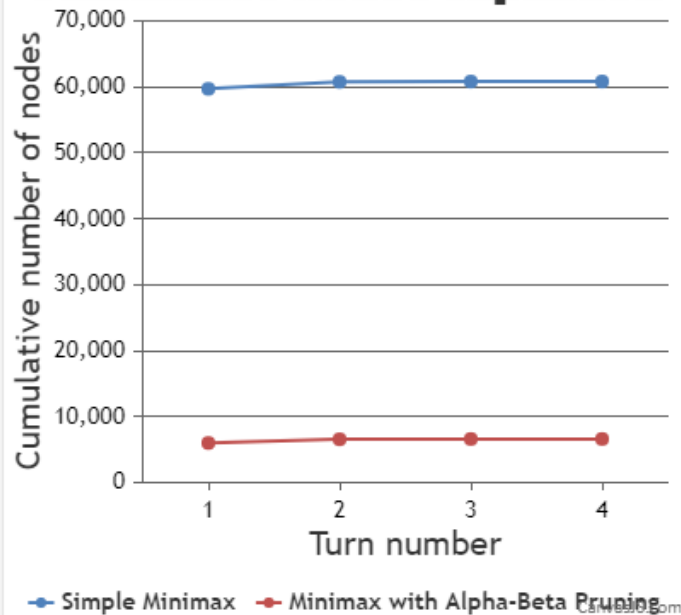
The time taken by Simple Minimax algorithm to take turn is: 0ms
The time taken by Minimax with Alpha-Beta Pruning algorithm to take turn is: 0ms

The number of nodes expanded by Simple Minimax algorithm is: 60810
The number of nodes expanded by Minimax with Alpha-Beta Pruning algorithm is: 6614

Time taken by algorithms to take turn



Cumulative Nodes expanded



Note that the charts shown here are displayed for depth = infinity, hence the entire game tree is searched.

It is clearly visible from the charts that Alpha-Beta pruning significantly optimizes the minimax search algorithm by reducing our search space.