P2 Intro:

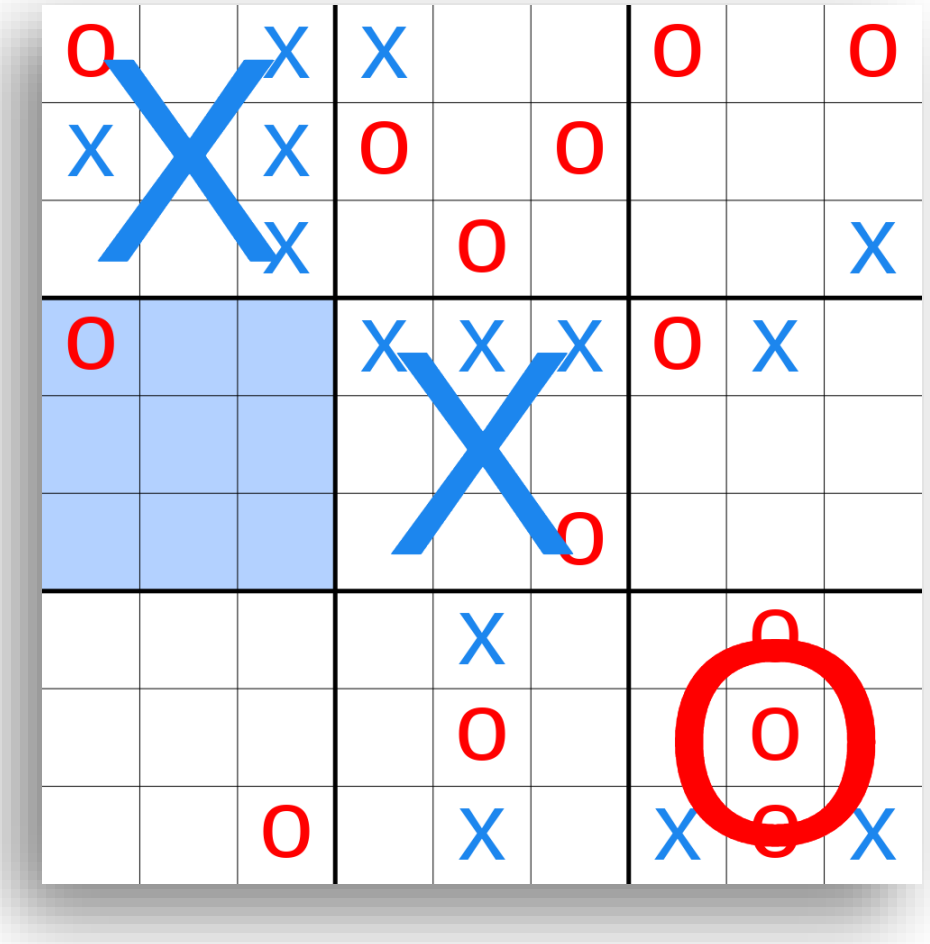
MCTS for Ultimate Tic Tac Toe

What is Ultimate Tic Tac Toe?

Tic Tac Toe:

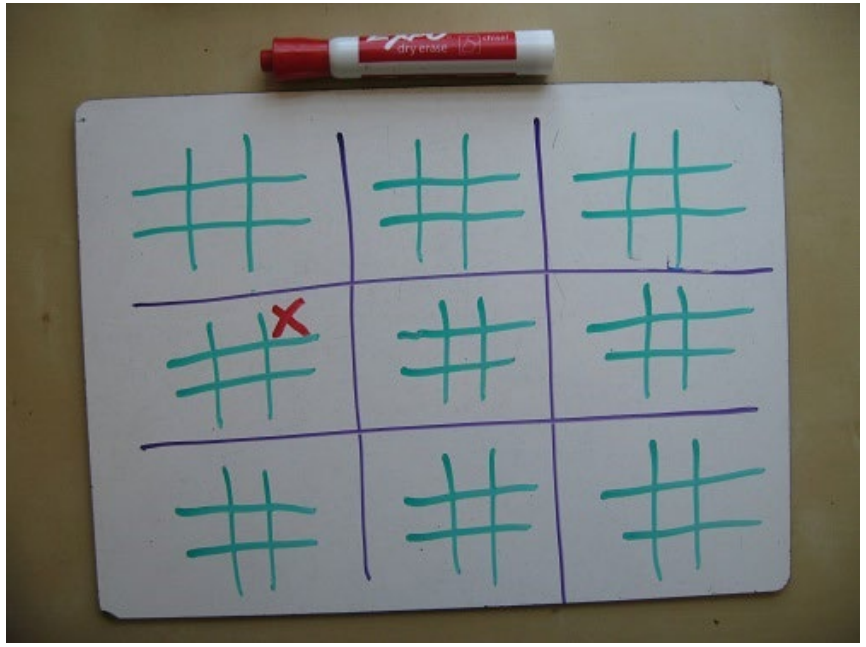
What is Ultimate Tic Tac Toe?

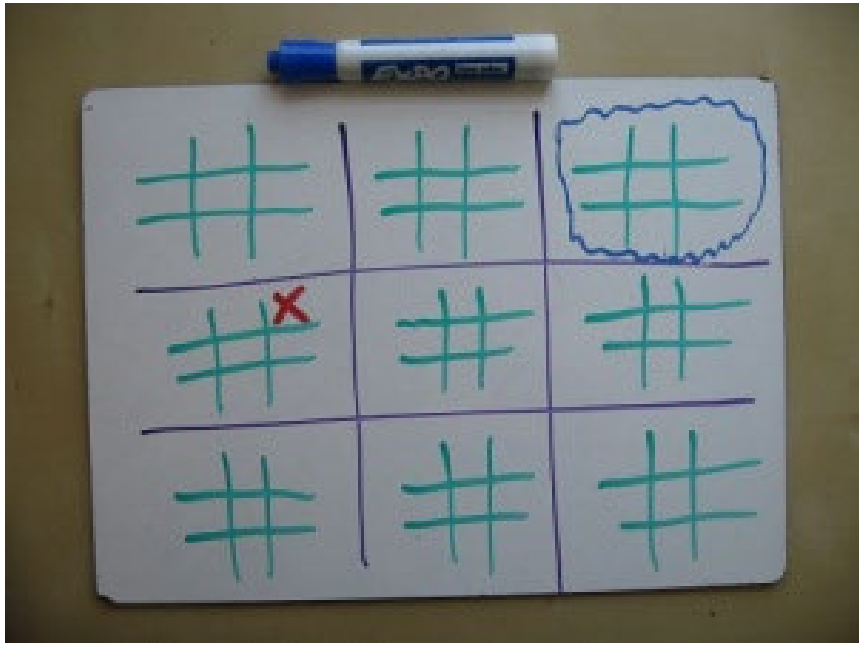
Ultimate Tic Tac Toe:



What is Ultimate Tic Tac Toe?

How to move:





The Programming Assignment

1. Implement basic MCTS bot (4 pts)
2. Implement modified MCTS bot (2 pts)
    - (refer to lecture and suggestions in .docx instructions)
3. Evaluate differences between your bots (4 pts)
4. (Extra credit) Do 3 again, using time rather than size of MC tree as constraint (# of pts TBD)

Provided files

- p2_play.py: Interactive command-line player.
  - “python p2_play.py human human” for human v human play
  - “python p2_play.py human rollout_bot” for human v robot play
  - Generally: “python p2_play.py P1 P2”
- p2_sim.py: Multi-game simulator with minimal output
  - E.g., “python p2_sim.py random_bot rollout_bot”
  - Generally: “python p2_sim.py P1 P2”

Note: Adding new bots to these scripts requires modifying the ‘players’ dictionary they define.

- p2_t3.py: Core simulation functions and Board class for ultimate tic-tac-toe. © 2014 Jeff Bradberry.

Provided files

- Sample bots:
  - random_bot.py: Picks and plays a random action.
  - rollout_bot.py: Picks and plays the action which, on average, tends to lead to the best outcome in random rollouts.

- Bots all define a think(board, state) function which returns an action.
  - You can ask the board for the legal actions in a state: *board.legal_actions(state)*

  - And you can ask the board for the next state after applying an action: *board.next_state(state, action)*

You code these files

- mcts_node.py: Utility class for MCTS nodes
- mcts_vanilla.py: Basic MCTS implementation
- mcts_modified.py: Modified/enhanced MCTS implementation

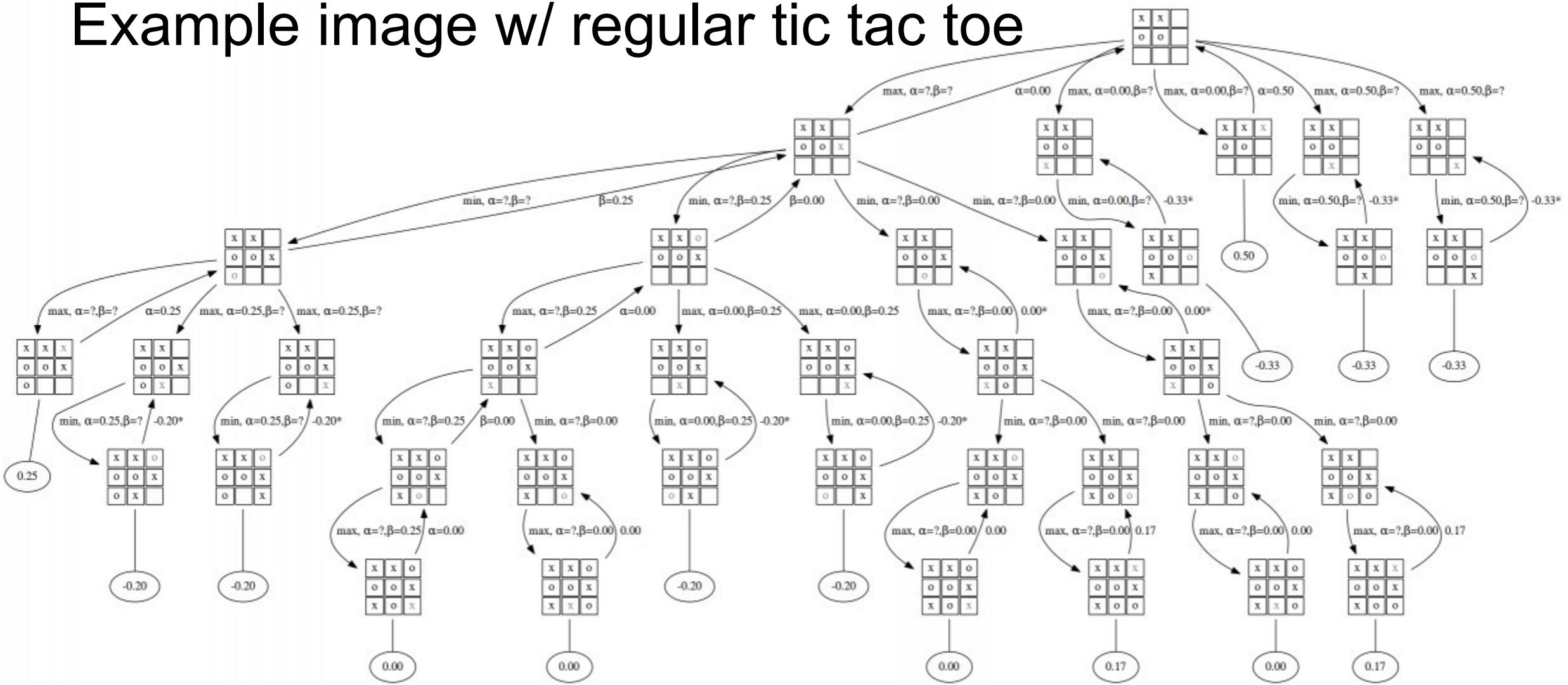
How to start (1)

- run p2_play.py a couple times with one or both human players, understand how p2_t3.py implements ultimate tic-tac-toe
- run p2_play.py with bots, understand how the two provided bots work
- run p2_sim.py with bots, understand how p2_sim.py works

How to start (2)

- implement MCTS bot
  - States are immutable tuples
  - Understand what “leaf nodes” are

Example image w/ regular tic tac toe



How to start (2)

- implement MCTS bot
  - States are immutable tuples
  - Understand what “leaf nodes” are
  - Backpropagate recursively

```
def backpropagate(node, score):
    #if node is root return
    #do update stats in node
    backpropagate(node.parent, score)
```

How to start (3)

- make sure the MCTS bot beats the random and random rollout bots
- implement the modified MCTS bot
- Run the evaluations
- (Optional) run the extra credit evaluations

Questions