- We are attempting to compute all possible unique states of a Tic-Tac-Toe game
- For simplicity, let's consider all rotations and symmetries as the same for now, which would save us time in calculating all of the possible boards and we can rotate all generated states once they are done
- For visualizing this, a graph is easier to use than a tree due to possible paths merging into the same state
- In Tic-Tac-Toe, there are 19,683 possible board configurations
- If we were to ignore symmetrical states, there are only 765 unique configurations
- Note that there are 8 symmetries per board, four rotations (0, 90, 18, and 270 degrees) and four mirrored versions (vertical mirror, and rotations of these)
- Let states be represented as a tuple of tuples, where each element is a tile on the board → 0 will represent an empty position, 1 will indicate an X, and 0 will indicate an O
- Using rotations and reflections, we will compute all unique symmetries for a given state, storing these in a list and running some sort of set function to ensure no duplicates
- Possible next states are generated by placing an X on an odd step and an O on an even step on empty tiles
- From there, new states are created by placing a marker on the board to indicate which player (X or O) went in a position
- Next, we verify if the current state is a game ending state (win/loss/tie)
- Note that ties take place when the board is full and no player formed a sequence of three pieces in a row
- We can use a DFS to explore all reachable game states, and for each one we will complete all possible next states, adding them to some file if they have not been processed already
- Note that there are nodes are shared between different game paths that lead to the same configuration, we need to avoid duplication
- In our construction, notes represent symmetry groups/unique configurations
- The DFS process starts from an empty board and iteratively computes next possible states, then it checks if a state has been seen before and, if so, it merges paths. Otherwise it creates a new node and continues traversal
- In the end, if we eliminate the symmetry and reflections there are only 765 distinct states, making it much easier to generate in our code and then we can process the reflections later
- This essentially reduces the possible configurations from 19,683 to 765 distinct states
- If we want to model this, we should use a graph to avoid redundancy that a tree would create, as it ensures each unique state is computed only once

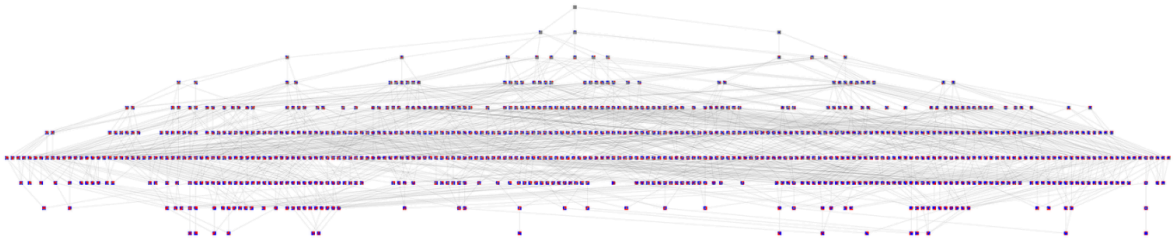

Figure 1: A diagram of all of the possible states of a tic tac toe board.