405 Assignment 3 Writeup

For this assignment we started by making the starting game functionality. Our main class Driver.java only starts the game and all of our logic is kept in other files and classes. To start, we implemented a user interface and then a game class. The game class controls the primary game functions such as changing players and checking for wins. This class also applies the move to the current game state and the user interface. It includes other essential functionality like validating plays and checking for a tie. It also notifies the player when it is his turn and accepts the users next move.

For the state space search and representing the current game state, we made a node class which represents the state of the game in a two-dimensional integer array. A zero represents an unplayed position while a one or two indicates a player or bot move. Included in the node class are many other methods that provide functionality like a special constructor to make a child node based on the parent node. This constructor had to be implemented in a special way to actually memory copy and not just pass by reference so that there were not overwriting other states. There is also a method getAndSetChildren() to get and set the children for a given node. For this class we overrode the equals() and compareTo() method so that we could use the Java sorting and equality functionality. But we had to use reverse logic so that the sort was in descending order. We also had to examine the underlying array, and not just the object to determine if one Node was equal to another Node.

The state space is represented as a class. StateSpace.java is a class that takes a given state or node, and makes the AI move based on the best first search. It returns only an integer to the game class. The search starts by generating the first children. It then assigns a heuristic value to each child. Our heuristic determines a bot win or a user win and places the same heuristic value for them if the user win is less than 3 levels down. This allows the search to block another player’s move as well as win. For the first 3 levels they all get the same heuristic value which forces the algorithm to expand all the children and examine them. After the first three levels the game expands children until it finds a winning state and then returns an integer. After level 3 we do not care about a user win and are only interested in finding a win for the bot. To be a considered a goal and end the best first search, the node must either be a bot winning state or a user winning state and be less than 3 levels down. In this class, there are other methods that check a node or game state for a win and return 0 for no win, 1 for a user win, and 2 for a bot win.

A close up of a logo

Description generated with very high confidence

A close up of a logo

Description generated with high confidence