Project 3: N Queens

Joseph Espy

Problem

Given an n*n chess board between n = 3 and n = 25, empty aside from a queen in each column, is there a way to move the queens such that no queen can attack another queen.

Solution

I use an iterative deepening search to find solutions. I start at an initial position and then explore the game tree until I find a position where there are no conflicts.

My iterative deepening algorithm keeps a stack of all of the positions I could move to by moving a single queen. When I expand a node, I look at all of the positions I can get to by moving one queen within her column. I check against a hash set of positions that I know I have already evaluated. This is a graph searching problem not a tree searching problem so I need to keep track of locations I have already seen. I then check if this node is correct (if all of the queen positions are nonattacking). I keep expanding nodes from the top of the stack until there are no more nodes (there is no solution for this value of n), or I have found a solution (I print the first winning board that I find).

The assignment sheet is unclear in it's instructions. It explicitly states that we should use an iterative search, which implies an uninformed search for the first valid state. This is what I implement. It also says to include a description of a heuristic for your problem. I show below the heuristic that I would use if I were not asked to use iterative deepening search, and instead asked to use A*.

The heuristic I would chose to inform my choice as I expand nodes is the number of equivalence classes of queens of size 2 or more. Because each equivalence class of size two or more requires at least one move, it must take at least as many moves as there are equivalence classes. So it is an admissible heuristic. An equivalence class of queens is defined as a set of queens that can attack another queen in the class. For instance, if queen 1 can attack queen 2, and queen 2 can attack queen 3, they are all part of the same equivalence class. I first expand boards with fewer equivalence classes. I use A* to search the tree. Without rote repeating of what the algorithm does, A* searches nodes that the heuristic believes are close to the solution. When it finds a path to the solution, it looks at only the nodes that the heuristic believes might be faster. If the heuristic is optimal, then it will find the shortest path. The assignment did not ask for the shortest path, it asked for a solution to the N Queens problem.