1 n-Queens

Max Friedrich William Beverl inveted the eight queens puzzle in 1848: place 8 queens on a chess board such that none of them can capture any other. The problem, and the generalized version with n queens, has been studied extensively (a Google Scholar search turns up over 3500 papers on the subject).

Queens can move any number of squares along rows, columns, and diagonals (left); An example solution to the 4-queens problem (right).

a) Formulate n-queens as a search problem, using the following state-space representation of: a set of boards, in which each space on the board may or may not contain a queen.

Start State: An empty board

Successor Function: Return all boards with one more queen placed anywhere

Goal Test: Returns True iff n queens are on the board such that no two can attack each other

b) How large is the state space in your formulation? $2^{n^2}$, or 18,446,744,073,709,551,616 for 8-queens.

c) One way to limit the size of your state space is to limit what your successor function returns. Reformulate your successor function to reduce the effective state-space size. The successor function is limited to return legal boards. Then, the goal test need only check if the board has n queens.

d) Give a more efficient state space representation. A more effective representation is to have a fixed ordering of queens, such that the queen in the first column is placed first, the queen in the second column is placed second, etc. The representation could be a n-length vector, in each each entry takes a value 1-n, or “null”.
2 Search algorithms in action

For each of the following graph search strategies, work out the order in which states are expanded, as well as the path returned by graph search. In all cases, assume ties resolve in such a way that states with earlier alphabetical order are expanded first. The start and goal state are S and G, respectively. Remember that in graph search, a state is expanded only once.

a) Depth-first search.
   States Expanded: Start, A, C, D, B, Goal
   Path Returned: Start-A-C-D-Goal

b) Breadth-first search.
   States Expanded: Start, A, B, D, C, Goal
   Path Returned: Start-D-Goal

c) Uniform cost search.
   States Expanded: Start, A, B, D, C, Goal
   Path Returned: Start-A-C-Goal

d) Greedy search with the heuristic $h$ shown on the graph.
   States Expanded: Start, D, Goal
   Path Returned: Start-D-Goal

e) $A^*$ search with the same heuristic.
   States Expanded: Start, A, D, B, C, Goal
   Path Returned: Start-A-C-Goal