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*
Max Friedrich William Bezzel invented 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? \(2^{n^2}, \text{ or } 18,446,744,073,709,551,616 \text{ for } 8\text{-queens.}\)

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) How large is the state space in your formulation with an efficient successor function? There are \(n^2\) choices for the first queen, \(n^2 - 1\) choices for the second queen, and so on. But, order doesn't matter. So we have \(\frac{n^2!}{(n^2-n)!n!}\). That's 4,426,165,368 possible boards for the 8-queens problem.

e) Give a more efficient state space representation. How large is the state space, with and without an efficient successor function? 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”. Without an efficient successor function, we have \(n\) choices for each queen, so the total state space is \(n^n\) (16,777,216 with 8-queens). Using an efficient successor function, for the first queen, we have \(n\) choices, for the second we have \(n - 1\), etc, so the total state space is \(n!\) (40,320 for 8-queens).
3 15-puzzle

The puzzle involves sliding tiles until they are ordered correctly. To solve these puzzles efficiently with A* search, good heuristics are important.

(1) Create a heuristic for the 15-puzzle based on the number of misplaced tiles.
*Count the number of tiles out of place, not including the blank tile.*

(2) Create a heuristic using Manhattan distance.
*Sum the Manhattan (city block) distances between each tile’s current position and its intended position.*

(3) Explain why your heuristics are admissible.
*These heuristics are both relaxations of the original problem so the heuristic estimate is always less than or equal to the actual cost of reaching the goal. Note that if you include the blank tile in the estimate, this heuristic is no longer admissible (think about the case where only 1 tile is out of place).*