Problem Set #1 Solutions

1. (15 points) Uninformed search
   (a) breadth-first search

   (b) depth-first search

   (c) iterative deepening search (starting with depth 0, and incrementing the depth by 1 in each iteration)

2. (20 points) Formulating search problems
   States: A cryptarithmetic puzzle with some (or none) of the letters assigned to digits.

   Operators: Only a single parameterized operator is necessary: \( \text{Assign}(\text{Letter}, \text{Digit}) \). This operator replaces all occurrences of a single letter (that is not already assigned a value) with a digit not already assigned to another letter in the puzzle.

   Initial state: A cryptarithmetic puzzle with none of the letters replaced by digits.

   Goal test: All the letters in the cryptarithmetic puzzle have been assigned distinct digit values and this assignment represents a correct sum for the puzzle.

3. (10 points) Search spaces
   Consider the search space shown below where every node has a branching factor of 5. The goal is at depth 4, and is on the far left hand side of the search tree, where the children of a node are expanded in left to right order.

   Depth first search will immediately find this by expanding only the 5 nodes on the path to the goal.
Iterative Deepening, on the other hand, will need to search the entire search tree down to level 0, then 1, then 2, then 3 and only when it searches down to level 4 will it find the goal node. With a branching factor of 5, that means that Iterative Deepening will need to expand $1 + (1 + 5) + (1 + 5 + 25) + (1 + 5 + 25 + 125) + 1 = 195$ nodes! This is 39 times as many nodes as Depth first search expands in this case.

If the goal node was at an even lower depth in the tree (but, still on the far left hand side), Iterative Deepening would perform even worse in comparison to Depth first search.

4. (20 points) A* Search
1. One admissible heuristic would be $h-hat = 0$.

A better one would be $h-hat = \text{the number of queens yet to be placed} = 4 - g$, where $g$ is the number of queens already placed.

An even better one would subtract some positive factor from $4 - g$ that depends on the number of cells in the grid that can have a queen legally placed in them. For example, we might use $h-hat = 4 - g - q/16$, where $q$ is the number of cells in the array that can have queens legally placed in them. Since, all else being equal, $q$ decreases with the depth in the tree, we might experiment with a function that counteracts that decrease, say, $h-hat = 4 - g - (g^2)(q/16)$, which is admissible since $h = 4 - g$ for paths leading to a goal, and $h$ is infinite for paths that do not lead to a goal.

2. See the diagram on the next page. Nodes are expanded in the order shown by the circled numbers. Cells with dots cannot legally have queens placed in them. Strictly speaking, A* does not terminate until the goal node is selected for expansion (in order to guarantee a shortest path), but in this case (since all paths to goal nodes are of length 4), we can terminate as soon as we reach a goal node.
5. (15 points) Termination of A*
When a goal node has been selected for expansion, A* has found the shortest path to it, but it might not have found the shortest path to a goal node still on OPEN (i.e., in the priority queue of nodes to be expanded). The following example illustrates this point. The numbers next to the nodes are the $g\text{-hat} + h\text{-hat}$ values, and the numbers next to the arcs are their (operator) costs. The goal node at the bottom has been placed on OPEN, but, later, when the rightmost node is expanded, a shorter path to it is found.
6. (20 points) Non-admissible heuristics

Our (slightly brain-damaged) non-admissible heuristic gives value 0 to every board configuration of the 8-puzzle where the 6 tile is not immediately above the 8 tile in the grid, and gives value 100 to every board configuration where the 6 tile is immediately above the 8 tile in the grid. Evidently, our heuristic thinks it is a very bad idea to have the 6 tile above the 8 tile.

This heuristic is clearly non-admissible since the following board configuration is only one step away from the goal, but our heuristic would guess that it is 100 steps from the goal (since the 6 tile is immediately above the 8 tile). Thus our h-hat > h, so it is non-admissible.

Consider the following search that finds a non-optimal path to the goal:

From this node there is a path to the goal in 1 step, but this node is not expanded. Note that all other nodes coming out of the initial state, except for the node at the left, have f-hat value of 101, since moving the 2 or 4 tiles in the initial state still leaves the 6 tile above the 8 tile. Thus, this is a good state that our inadmissible heuristic function makes to “look bad”.

On the left we show a non-optimal found that will be found prior to the state above, that we know is on the optimal path, ever being expanded.
Continuing the search from the last node we left off at:

\[
\begin{array}{c}
\text{f-hat } = 0 + 4 = 4 \\
\begin{array}{ccc}
1 & 2 & 3 \\
4 & 8 & \\
7 & 6 & 5 \\
\end{array}
\end{array}
\]

\[\text{Left} \]

\[
\begin{array}{c}
\text{f-hat } = 0 + 5 = 5 \\
\begin{array}{ccc}
1 & 2 & 3 \\
4 & 8 & \\
7 & 6 & 5 \\
\end{array}
\end{array}
\]

\[\text{Up} \]

\[
\begin{array}{c}
\text{f-hat } = 0 + 6 = 6 \\
\begin{array}{ccc}
1 & 2 & 3 \\
4 & 8 & 5 \\
7 & 6 & \\
\end{array}
\end{array}
\]

\[\text{Right} \]

\[
\begin{array}{c}
\text{f-hat } = 0 + 7 = 7 \\
\begin{array}{ccc}
1 & 2 & 3 \\
4 & 8 & 5 \\
7 & 6 & \\
\end{array}
\end{array}
\]

\[\text{Down} \]

\[
\begin{array}{c}
\text{f-hat } = 0 + 8 = 8 \\
\begin{array}{ccc}
1 & 2 & 3 \\
4 & 5 & \\
7 & 8 & 6 \\
\end{array}
\end{array}
\]

\[\text{Left} \]

\[
\begin{array}{c}
\text{f-hat } = 0 + 9 = 9 \\
\begin{array}{ccc}
1 & 2 & 3 \\
4 & 5 & \\
7 & 8 & 6 \\
\end{array}
\end{array}
\]

\[\text{Up} \]

\[
\begin{array}{c}
\text{f-hat } = 0 + 9 = 9 \\
\begin{array}{ccc}
1 & 2 & 3 \\
4 & 5 & 6 \\
7 & 8 & \\
\end{array}
\end{array}
\]

\[\text{Goal state} \]