Graph Search

Consider the "eight puzzle". There are eight tiles numbered 1 through 8 on a 3 by three grid with nine locations so that one location is left empty. We can "move" by sliding a tile adjacent to the empty location into the empty location. The goal is to take a given configuration and through a series of moves arrange the tiles in order from left to right and top to bottom. The set of configurations of the tiles forms a graph where two nodes (configurations) are connected if one node can be converted to the other by a single move.

We are interested in computing the shortest path from a given start node to a given goal node.

Dijkstra Shortest Path

We are given a graph G where each edge is assigned a non-negative weight and we are also given a start node s and goal node g. We want the path of least weight from s to g.

1. Initialize Q to be the set containing the single pair $\langle s, 0 \rangle$.
2. Initialize S to be the empty set.
3. If Q is empty terminate with failure (there is no path from s to g).
4. Remove a pair $\langle n, w \rangle$ from Q with minimum weight w (over all elements of Q).
5. If S already contains a weight for n, i.e., if S contains $\langle n, w' \rangle$ for some w', then continue again from step 3.

6. If $n = g$ then terminate and output w as the cost of the shortest path from s to g.

7. Otherwise, add $\langle n, w \rangle$ to S.

8. For each edge $\langle n, m \rangle$ with weight w' in the graph G add the pair $\langle m, w + w' \rangle$ to Q.

9. repeat from step 4.

Note that if removing $\langle n, w \rangle$ form Q results in adding $\langle m, w' \rangle$ then because edge weights are non-negative we have that $w' \geq w$. This implies that the weights removed from Q monotonically increase. This implies that when a pair $\langle n, w \rangle$ is added to S, w is the weight of the shortest path to n.

Admissible Heuristics

Let h be a function form the nodes of G to non-negative real numbers.

The function h is admissible if, for any node n, the distance from n to g is at least $h(n)$.

The function h is called *monotone* if for any edge $\langle n, m\rangle$ with weight w we have $|h(m) - h(n)| \leq w$

Suppose h is monotone and $h(g) = 0$. Consider a path from g to n. As we cross an edge moving from g to n along this path the increase in h can be no larger than the weight of the edge. So $h(n)$ cannot be larger than the weight of the path. This gives us that if $h(g) = 0$ and h is monotone then h is admissible.

Manhattan distance in the eight puzzle is both monotone and admissible (this is the sum over tiles of the number of rows plus the number of columns that the tile must be moved).

A*

1. Initialize Q to be the set containing the single pair $\langle s, 0 \rangle$.
2. Initialize S to be the empty set.
3. If Q is empty terminate with failure (there is no path from s to g).
4. Remove a pair $\langle n, w \rangle$ from Q minimizing $w + h(n)$.
5. If S already contains a weight for n, i.e., if S contains $\langle n, w' \rangle$ for some w' then continue again from step 3.
6. If $n = g$ then terminate and output w as the cost of the shortest path from s to g.
7. Otherwise, add $\langle n, w \rangle$ to S.

8. For each edge $\langle n, m \rangle$ with weight w' in the graph G add the pair $\langle m, w + w' \rangle$ to Q.

9. repeat from step 4.

If h is monotone then the sum $w + h(n)$ of the item removed from the queue monotonically increases. This implies that when we remove $\langle n, w \rangle$ from the queue, the weight w is the least possible weight for n. Note that monotonicity alone gives correctness of the algorithm — we do not need that $h(g) = 0$. By taking the distance to goal into account A* can be vastly more efficient than Dijkstra Shortest path.