The Complexity

The complexity of the algorithm is dictated by the amount of edges that are to be placed. We will refer to this number as N. Because we hardcode a path of 6 to begin with, we have (N – 6) edges to place. For each edge, 8 different possible paths are tested to see if they're legal. This means that currently we have a complexity of 8(N – 6). After each edge is chosen, the Edmonds-Karp algorithm is ran on it, which in itself has a complexity of N^2 in the worst case. Because this is done after each edge is placed, the N^2 algorithm is ran 8(N – 6) times, which comes up to 8N^2(N – 6).

Because complexity is dictated by the input size, and the input can grow arbitrarily large, the constants are dropped off our analysis, since they will not be influencing the outcome by much. This leaves us with a complexity of N^2(N), which simplifies to O(N^3).

The complexity only depends on the amount of edges to be placed, and not on the size of the grid, since increasing the grid's size does NOT add more legal moves or any extra computations. This is because our algorithm ONLY places edges that keep the graph connected, which means going off into the grid away from the sink will not happen. Also, since we don't remove edges due to the greedy nature of our algorithm, all placements are final. So our diagnosis of O(N^3) is not influenced by the size of the graph.

The algorithm is still very fast, because N is incredibly small. For this particular problem, we found that an N^3 algorithm that acts greedily is an efficient way of solving this problem.