Assignment Zero Report

Elements of Artificial Intelligence
Ninaad Joshi | ninjoshi@iu.edu

---

1. Abstraction used :

- The state space is the set of all possible states achievable from the initial state, regardless of being valid or invalid, in this case, a chessboard of size N x N where the N rooks are placed at all the possible positions, where N is the number of rooks on the chess board . (Denoted by S)
- The initial state is the starting state which, in this scenario, is denoted by an empty chess board. (Denoted by S_0)
- The successor function is a function which provides a set of all achievable states from a given input state. In this case, the successor function provides the states where the rooks are placed on any square of the chessboard. (Denoted by SUCC(s) where s is the input state)
- The goal state is a state or a set of states which is the set of solutions for the given problem. In this case, the goal states are when N rooks are placed on the chessboard with no conflicts between any of the rooks.
- Cost function is a function that calculates the cost of calculating the expense of a particular move. In this case, the cost function is a uniform cost function i.e. the cost of transition from a state to any of its successor states is the same. (Denoted by COST(s_1 -> s_2) where s_1 is the current state and s_2 is the next state)

<u>Mathematical model</u> :

S = { a chessboard of size N x N where the N rooks are placed at all the possible positions, where N is the number of rooks on the chess board }

S_0 = { state of board such that there is no piece placed on the chess board }

SUCC(s) = { set of all states achievable from given input state s }

G = { set of states in which N rooks are placed on the chessboard with no conflicts between any of the rooks }

COST(s_1 -> s_2) = cost of transition from s_1 to s_2, which is same for all states in this case.

This explains the abstraction used in the given program.

2. Modifications performed :

- Modification to the original state space was done to reduce the state space by considering only the valid states. Thus, new state space omits the states with conflicting rooks.

S = { a chessboard of size N x N where the N rooks are placed at all the possible valid positions where there are no conflicts between the placed rooks, where N is the number of rooks on the chess board }

- The successor function was modified to output all the valid set of states where there is no conflict between any of the placed rooks.

SUCC(s) = { set of all valid states achievable from given input state s such that no conflict arises between the placed rooks }

3. BFS to DFS :

- To switch the algorithm from DFS to BFS, a change was made to the Fringe/Frontier data structure.
- The Depth First Search algorithm uses a stack for storing the output states from the successor function following the Last In First Out principle.
- Thus, a queue was used instead for converting the algorithm to Breadth First Search which uses the First In First Out principle for storing the states.

4. Explain the Behavior :

- Below is a table containing the output times required for running the BFS and DFS algorithms for finding a goal state. The transition from DFS to BFS results in significant changes in the running time of the program. The DFS algorithm checks the child states at a depth rather than checking all the child states of the parent state. Thus, it performs faster if a solution can be found in a particular subtree of the state tree. Thus, it performs significantly faster because it does not need expand all the nodes of the tree to check for a solution.

Number of rooks N	Depth First Search	Breadth First Search
1	0m0.008s	0m0.008s
2	0m0.009s	0m0.010s

3	0m0.010s	0m0.010s
4	0m0.004s	0m0.008s
5	0m0.014s	0m0.018s
6	0m0.020s	0m0.009s
7	0m0.099s	0m0.009s
8	0m1.593s	0m0.009s
9	1m20.945s	0m0.010s

Thus, in this example, DFS provides a solution faster than BFS.