CSCI-B551: Elements of AI
Assignment 0

MOHIT SARAF
Sarafm@iu.edu

1. Set of Valid States: Arrangement of 0 to N Rooks on the Chessboard can be defined as a set valid state

The successor function: Adds a valid rook to the position where it is not attacked by already placed rooks and returns the board configuration.

The cost function: The cost function is irrelevant over here so we can consider it to be uniform.

Goal State: Arrangement of N rooks on the board such that no rook is attacked by any other rook.

Initial State: No rooks on the board

2. A successor function has been created successors2() where the function checks whether the Rooks already added are less than N to avoid generating states for N+1 Rooks. It also avoids generating successor states for the states where the rook is already present at a position.

3. To switch the code from DFS to BFS we will change the behavior of the fringe from stack to Queue by replacing successors2 (fringe.pop ()) with successors2 (fringe.pop (0)).

4.

N=	2	3	4	5	6
DFS Time	0.015s	0.013s	0.013s	0.014s	0.015s
BFS Time	0.015s	0.016s	0.019s	0.17s	20.168s

For N=2 to 5 both BFS and DFS have comparatively same performance but as the value of N increases it increases the state space exponentially due to which BFS performs slower than DFS as BFS explores all the branches at a depth before proceeding to the next one whereas DFS first explores a single branch to its full depth before proceeding to the next one. Also, BFS looks for the complete solution.

5. To implement this a value of 0.1 has been assigned to the unavailable position given as command line input which signifies the 'X' or the unavailable position on the board. For this problem we also created two new successor functions successorsNrooks() and successorsNqueen() and made sure we don't place a rook or queen at the unavailable .
In successorsNrooks() we check whether the Rooks already added are less than N to avoid generating states for N+1 Rooks. It also avoids generating successor states for the states where the rook is already present at a position. In successorsNqueen() we check whether the Queens already added are less than N to avoid generating states for N+1 Queens. It also avoids generating successor states for the states where the Queen is already present at a position. We also check for states where a new Queen is attacked by previous one horizontally, vertically and diagonally.

CSCI-B551: Elements of AI
Assignment 0

MOHIT SARAF
Sarafm@iu.edu

References:

- *Norvig and Russell, Artificial Intelligence: A Modern Approach,3rd ed., Prentice Hall, 2009.*
- *https://docs.python.org/2/library/functions.html#zip*