Alex Lu 03/05/2025

ESE 561 Artificial Intelligence
Assignment 1 Supplementary Information

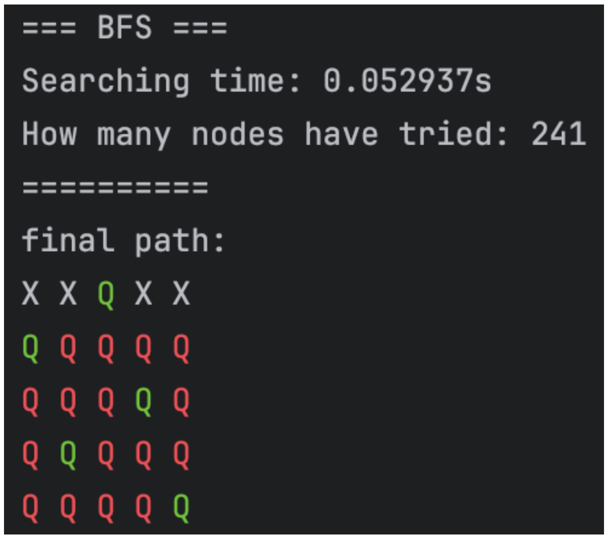
Background:

Recall from my previous analysis report: to solve the N-Queens problem, I utilized BFS, DFS, UCS, and A* separately. First, the user inputs the number of Queens to be solved and specifies the starting position in row 1. Imagine the solving process as a tree, where the root has only one Queen placed on the board at a position chosen by the user. Queens are placed row by row, and only positions that result in a valid placement are considered as nodes. In other words, the root can expand into nodes where a Queen is validly placed in the second row, and so on. Therefore, every node represents a valid state.

The tree depth corresponds to the number of Queens that have already been placed on the board. The results show that with this tree structure and node expansion mechanism, the N-Queens problem can be successfully solved. However, the heuristic function in A* is defined as the number of Queens that have yet to be placed on the board plus the number of conflicting Queen pairs. Since every node already represents a valid state, the heuristic function does not effectively differentiate between nodes. As a result, the performance of A* is indistinguishable from that of BFS.

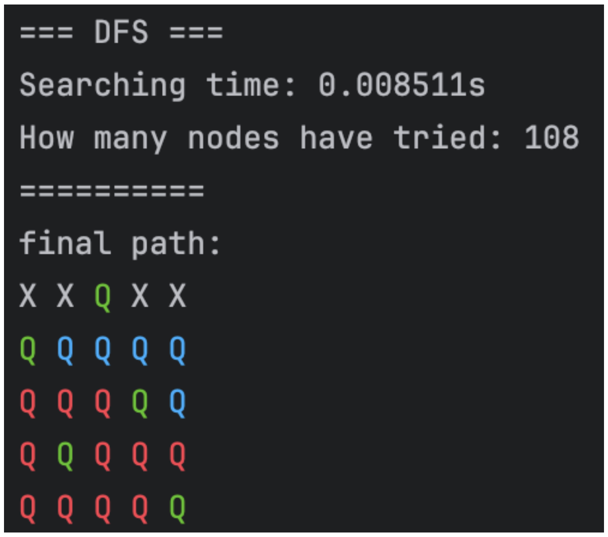
Optimization:

To highlight the performance of A*, I redesigned the rule for expanding nodes so that instead of only valid nodes being expanded, all nodes are expanded regardless of their validity. As a result, the heuristic function of A* can better distinguish which nodes are more promising and more likely to lead to the final solution. Using the 5-Queens problem and starting from column 3 of row 1 as an example, the results are shown in Figures (1), (2), (3), and (4). The green Q represents the placed position, the red Q represents the visited position, and the blue Q represents an expanded position that was not visited. From these images, we can easily see that A* is more efficient than BFS, DFS, and UCS. After modifying the node expansion mechanism, A* only needs to explore 8 nodes to reach the final solution, whereas BFS needs to explore 241 nodes. As a result, the heuristic function of A* plays a significant role and performs effectively.



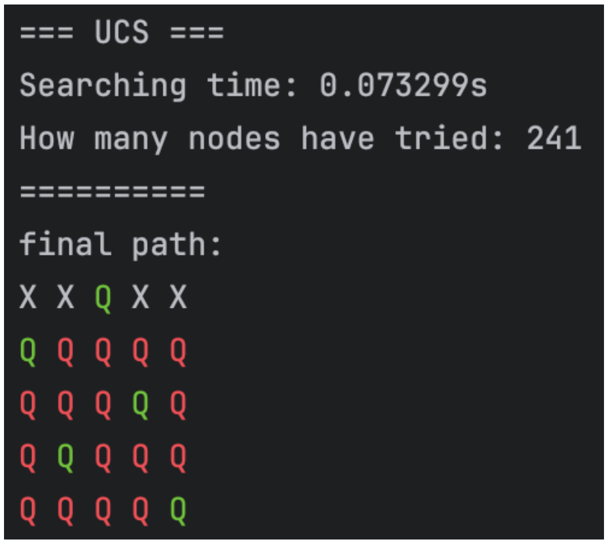
```
=== BFS ===
Searching time: 0.052937s
How many nodes have tried: 241
==========
final path:
X X Q X X
Q Q Q Q Q
Q Q Q Q Q
Q Q Q Q Q
Q Q Q Q Q
```

Figure 1: BFS



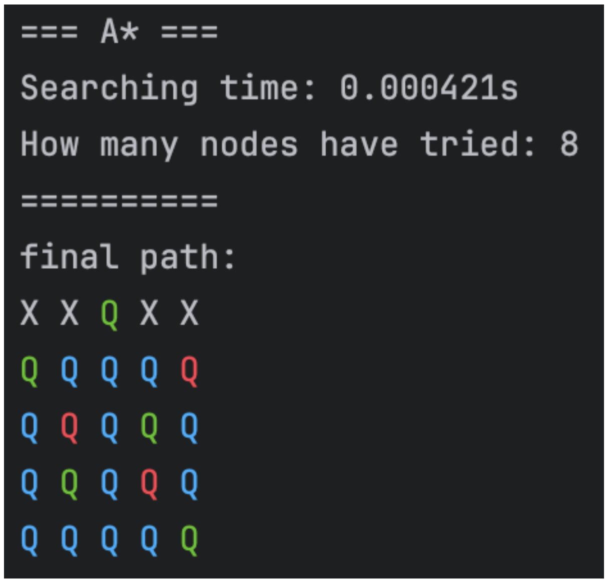
```
=== DFS ===
Searching time: 0.008511s
How many nodes have tried: 108
==========
final path:
X X Q X X
Q Q Q Q Q
Q Q Q Q Q
Q Q Q Q Q
Q Q Q Q Q
```

Figure 2: DFS



```
=== UCS ===
Searching time: 0.073299s
How many nodes have tried: 241
==========
final path:
X X Q X X
Q Q Q Q Q
Q Q Q Q Q
Q Q Q Q Q
Q Q Q Q Q
```

Figure 3: UCS



```
=== A* ===
Searching time: 0.000421s
How many nodes have tried: 8
==========
final path:
X X Q X X
Q Q Q Q Q
Q Q Q Q Q
Q Q Q Q Q
Q Q Q Q Q
```

Figure 4: A*