Methodology

I ran each search algorithm with test case one until it provided the correct result in a reasonable time. As I optimized the algorithms, I used larger and larger test cases. The only parameters for each algorithm are the starting and goal states. I did not have time to finish my implementation of A*, so it is not included. For DFS, there is no depth limit. For IDS, the depth limit is the largest available integer in c++, which is 2,147,483,647. Each algorithm uses std::unordered_set to keep track of the explored list, because it is implemented as a hash table which provides O(1) lookups in the general case.

Results

Search	Test Case 1		Test Case 2		Test Case 3	
	# of solution nodes	# of generated nodes	# of solution nodes	# of generated nodes	# of solution nodes	# of generated nodes
BFS	11	16	35	63	387	978
DFS	11	16	35	56	391	1152
IDS	11	16	35	56	391	1152

Discussion

The observed results are somewhat surprising. In every test case, DFS and IDS perform identically, which I suspect is an indication that IDS was implemented incorrectly. For the first test case, all three algorithms found the same solution and expanded the same number of nodes. For the second test case, all three algorithms agreed on the solution cost, however BFS expanded a few more nodes. Surprisingly, BFS outperforms both other searches on the final test case both finding a better solution and using fewer nodes to do so.

Conclusion

Based on these results, Breadth-first search seems to be the best uninformed search algorithm, although I would expect A* to have performed better had I implemented it. I would have also expected IDS to outperform the other two, but this was not the case based on these inputs.