Sudoku Project PDF Notes

Original:
https://www.researchgate.net/publication/358642884_Comparison_Analysis_of_Breadth_First_Search_and_Depth_Limited_S

- Algorithms to be used to solve a Sudoku puzzle:
  - Breadth First Search (BFS) - used to explore or search through a graph or tree level by level. It starts at the root (or starting node) and explores all its neighbors before moving to the next level of nodes. It uses a **queue** to keep track of nodes to visit and ensures the shortest path (in terms of edges) is found in an unweighted graph.
  - Depth Limited Search (DLS) - limits the depth of the search to a predefined maximum level. It explores nodes as far as the specified depth allows and backtracks when it reaches the limit. This prevents the search from going too deep and potentially getting stuck in infinite loops in cyclic graphs.
- DLS is more efficient and faster than BFS.
- If a Sudoku question has more than one answer, BFS will find it
- Sudoku - consists of 81 squares of 9 columns and 9 rows. Further divided into 9 partial squares that consist of 3×3 squares. Players fill the boxes with numbers from 1 to 9 but there cannot be repetition in one column and in one row
- BFS Advantage/Disadvantage:
  - ADV: If there is one solution then BFS will find it, and if there is more than one solution then the minimum solution will be found
  - DISADV: It requires a lot of memory, because it stores all nodes in one tree, and it takes a long time because it will test *n* levels to get a solution at the (*n* + 1) level.
- DLS Advantage/Disadvantage:
  - ADV: Quickly reaches the depth of the search space without wasting time on “shallow” states. It’s effective in branching search spaces since it avoids evaluation all nodes at a given level. Uses minimal memory, storing only nodes on the active path
  - DISADV: Needs to predefine the maximum depth, if the limit is too small then DLS cannot find the existing solution

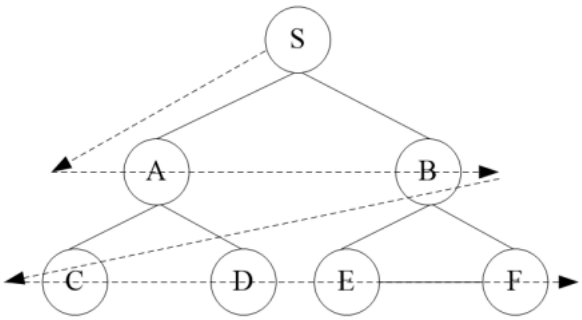


Figure 2. BFS algorithm

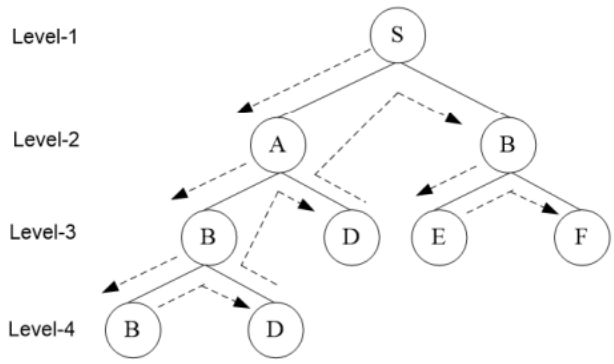


Figure 3. DLS algorithm

Example Sudoku problem

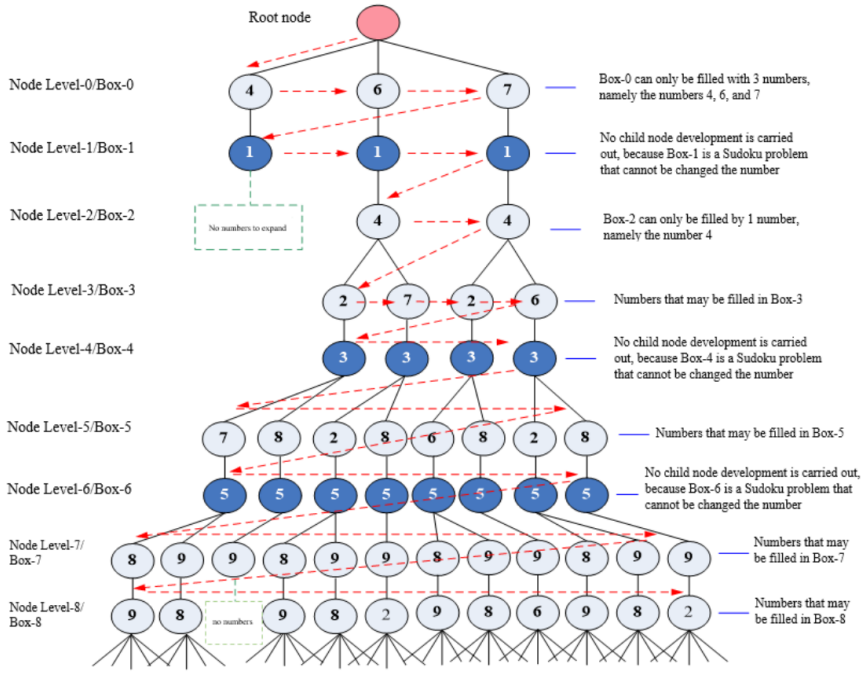
	1			3		5		
	3		9		4	7		
8		9				4		
		6					7	
2			8		1			5
	8					2		
		7				8		4
		8	1		9		2	
		2		5			6	

Figure 5. Sudoku game questions

BFS Method

1. Search starts from an empty root node
2. From this root node, all possible numbers that can be entered in box-0 are developed without violating the Sudoku rules. Each of these possible numbers is entered into a single node at level-0.
3. If the next box is a Sudoku question box (a number that cannot be changed when working on a Sudoku problem), then the parent node does not need to develop child nodes and all parent nodes have the same child node, namely the node that contains the numbers in the question box.
4. Meanwhile, if the next box is the answer box, then develop all possible numbers that can be entered in the box without violating the Sudoku rules.
5. Continue developing and searching for nodes until level 80 is reached. The answer is a series of steps from node level-0 to node level-80.
6. If none of the nodes can reach the 80th level, then it is certain that the Sudoku problem has no solution.

TLDR: Starts from top-left box, continues through the row, puts possible numbers in the box goes to next box. Then starts at next row and continues the process until box 80 (bottom-right) is reached.



DLS Method

1. The search starts from an empty root node.
2. From the root node, 1 possible number will be developed that can be entered in box-0 and does not violate the Sudoku rules. This number is a level-1 child node. Furthermore, the child nodes are developed from level-1, and so on.
3. If no child nodes can be expanded, because all numbers violate Sudoku rules, then backtrack to the parent node and develop another child node.
4. If the next box is a Sudoku problem (a number that cannot be changed when working on a Sudoku problem), then the parent node does not need to develop a child node and goes directly to the child node containing the Sudoku question and develops other child nodes.
5. Continue searching and expanding nodes until you reach the 80th level node. Sudoku's answer is a series of steps from node level-1 to node level-80.
6. If there is a backtrack on the root node, it can be ascertained that the Sudoku question does not have an answer.'

TLDR: Starts from the top-left box, puts possible numbers and makes them parents, the next box will be children for those parents. If another box has a possibility to be the same number as a parent - that parent will no longer have children. Continue through with each tree.

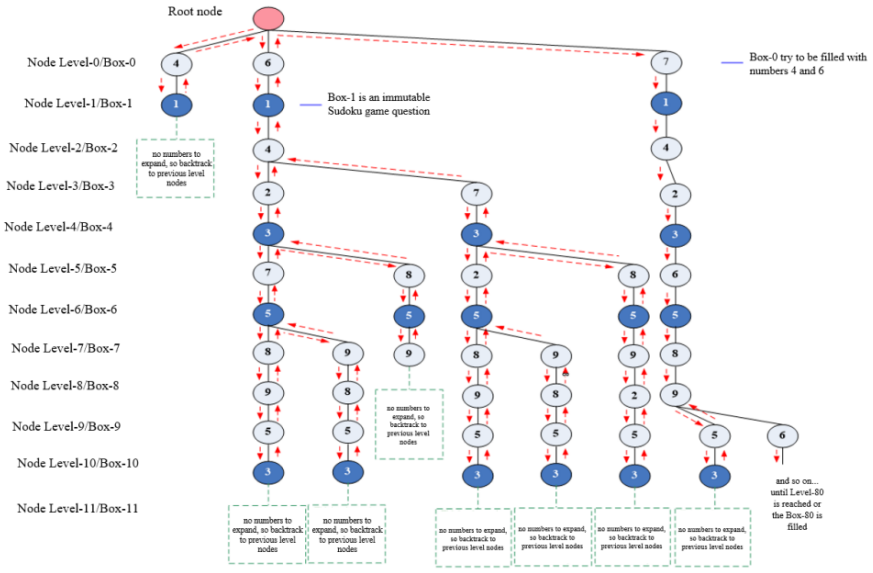


Figure 7. Sudoku game answer search process using DLS

Conclusion:

BFS requires expanding more nodes than DLS to find a solution because BFS systematically explores all possible numbers for each box, starting from the first box to the last. In contrast, DLS prioritizes depth-first exploration by placing a single number in a box, moving immediately to the next box, and continuing this process until all boxes are filled. If a rule violation occurs, DLS backtracks to the previous box, changes the number, and resumes the search.

Due to BFS exploring a larger number of nodes, it is less efficient than DLS in terms of time and resource usage for solving Sudoku. However, BFS has the advantage of being more structured and thorough, capable of finding all possible solutions if multiple answers exist. Overall, DLS is more efficient than BFS for finding solutions to Sudoku, requiring fewer nodes and less time.