**Summary of Hybrid BFS-DLS Improvement**

At the start of our project, we implemented a basic 9x9 Sudoku solver using BFS and DLS separately. However, we quickly realized that each algorithm had its drawbacks: BFS used too much memory for larger or more complex puzzles, and DLS could fail if the depth limit wasn’t set right. That’s when we decided to combine the strengths of both algorithms into a hybrid approach—Hybrid BFS-DLS.

This hybrid method dynamically adjusts the depth limit, switching between the exhaustive exploration of BFS and the efficient, depth-focused approach of DLS. It’s designed to find a solution efficiently, without overwhelming memory, while still maintaining enough exploration to solve puzzles of varying complexity.

**Does it work on any puzzle or only sometimes?**

The hybrid algorithm is built to work on almost any standard Sudoku puzzle. It adapts to the puzzle’s difficulty and adjusts its approach dynamically, making it more reliable across different types of puzzles. While BFS can sometimes miss solutions for harder puzzles due to memory issues, and DLS can struggle with very restrictive depth limits, the hybrid approach balances both issues. So yes, it works most of the time.

**Is it efficient time- or space-wise?**

The hybrid algorithm is both time- and space-efficient compared to using BFS or DLS alone. It uses BFS to explore the puzzle broadly, but it switches to DLS for deeper exploration once it narrows down potential paths. This saves memory, as BFS would otherwise consume a lot of resources by storing all the nodes at each depth level. On the time side, since the algorithm can focus its search dynamically based on puzzle complexity, it’s faster than BFS for harder puzzles while avoiding the inefficiencies of DLS.

**Is it better to find a solution fast or to find multiple solutions?**

The primary goal of the hybrid algorithm is to find a single valid solution efficiently. While BFS can be set to find multiple solutions, the hybrid approach is optimized for finding just one solution, as that’s usually sufficient for Sudoku. However, the modular nature of the algorithm means that if you wanted to find multiple solutions, you could easily tweak it to do that. The hybrid method excels at finding that solution quickly, which is ideal for most use cases.

**Why**?

The reason the hybrid works so well is because it combines breadth-first exploration (to get a broad view of possible solutions) with depth-limited exploration (to dig deeper into promising paths). This combination means that the hybrid algorithm doesn’t get stuck like DLS might with small depth limits, and it avoids the excessive memory usage of BFS, which tries to explore everything at once. This dynamic adjustment allows the hybrid approach to handle both easy and tough puzzles efficiently.

For example, when solving a typical 9x9 Sudoku puzzle, our hybrid approach found 37 solutions in just 36 ms using BFS and then took 218 ms in total for the Hybrid BFS-DLS approach to find those solutions. This shows the clear advantage of using the hybrid model when it comes to both time and memory efficiency, especially when the puzzle is more complex. Overall, the hybrid algorithm’s ability to dynamically adjust to the problem’s needs makes it a solid tool for effectively solving Sudoku puzzles.

A screenshot of a puzzle

Description automatically generated