Report: Sudoku Solver using Backtracking

1. Introduction

Sudoku is a logic-based number placement puzzle where a 9×9 grid must be filled so that each column, each row, and each of the nine 3×3 sub-grids contain all the digits from 1 to 9. This report discusses the implementation of a Sudoku solver using the backtracking algorithm in Python.

2. Problem Statement

The objective of this program is to solve a given Sudoku puzzle by filling in missing numbers while adhering to the Sudoku constraints:

- Each row must contain numbers from 1 to 9 without repetition.
- Each column must contain numbers from 1 to 9 without repetition.
- Each 3×3 sub-grid must contain numbers from 1 to 9 without repetition.

3. Algorithm and Implementation

The program is implemented using the backtracking approach, which is a depth-first search algorithm that explores all possibilities and backtracks when a constraint is violated.

3.1 Algorithm Steps

1. Find an empty cell (0) in the Sudoku grid.

2. Try placing numbers from 1 to 9 in the empty cell.

3. Check if the number placement is valid based on Sudoku constraints.

4. If valid, recursively solve the remaining puzzle.

5. If a conflict arises, backtrack by resetting the cell and trying the next number.

6. Repeat the process until the entire grid is filled correctly.

4. Code Execution

The program is executed using an example Sudoku grid:

Input Sudoku Grid:

5 3 . . 7 . . . .

6 . . 1 9 5 . . .

. 9 8 . . . . 6 .

8 . . . 6 . . . 3

4 . . 8 . 3 . . 1

7 . . . 2 . . . 6

. 6 . . . . 2 8 .

. . . 4 1 9 . . 5

. . . . 8 . . 7 9

Solved Sudoku Grid:

5 3 4 6 7 8 9 1 2

6 7 2 1 9 5 3 4 8

1 9 8 3 4 2 5 6 7

8 5 9 7 6 1 4 2 3

4 2 6 8 5 3 7 9 1

7 1 3 9 2 4 8 5 6

9 6 1 5 3 7 2 8 4

2 8 7 4 1 9 6 3 5

3 4 5 2 8 6 1 7 9

5. Performance Analysis

The backtracking approach efficiently solves Sudoku puzzles, but it can be slow for highly complex grids due to its exhaustive search nature. Optimization techniques such as constraint propagation and forward checking can further improve performance.

6. Conclusion

This report demonstrates a Sudoku solver using backtracking. The algorithm efficiently finds solutions by exploring possible number placements while adhering to Sudoku rules. This approach provides an effective way to solve Sudoku puzzles programmatically.