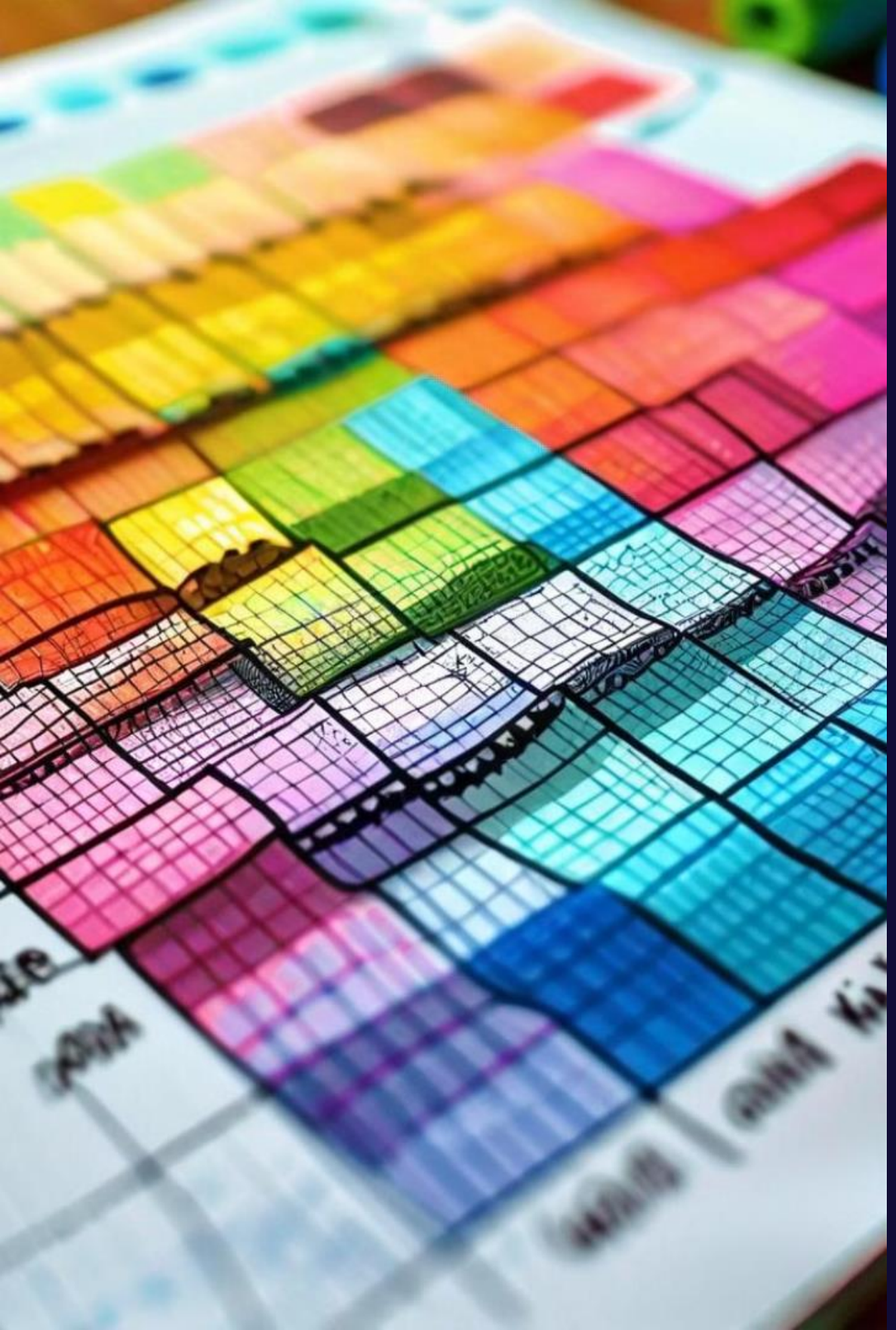


GT Mini Project

Sudoku Solver using Graph Coloring

Presented By Group-5

Debasish Dey(2202040017)

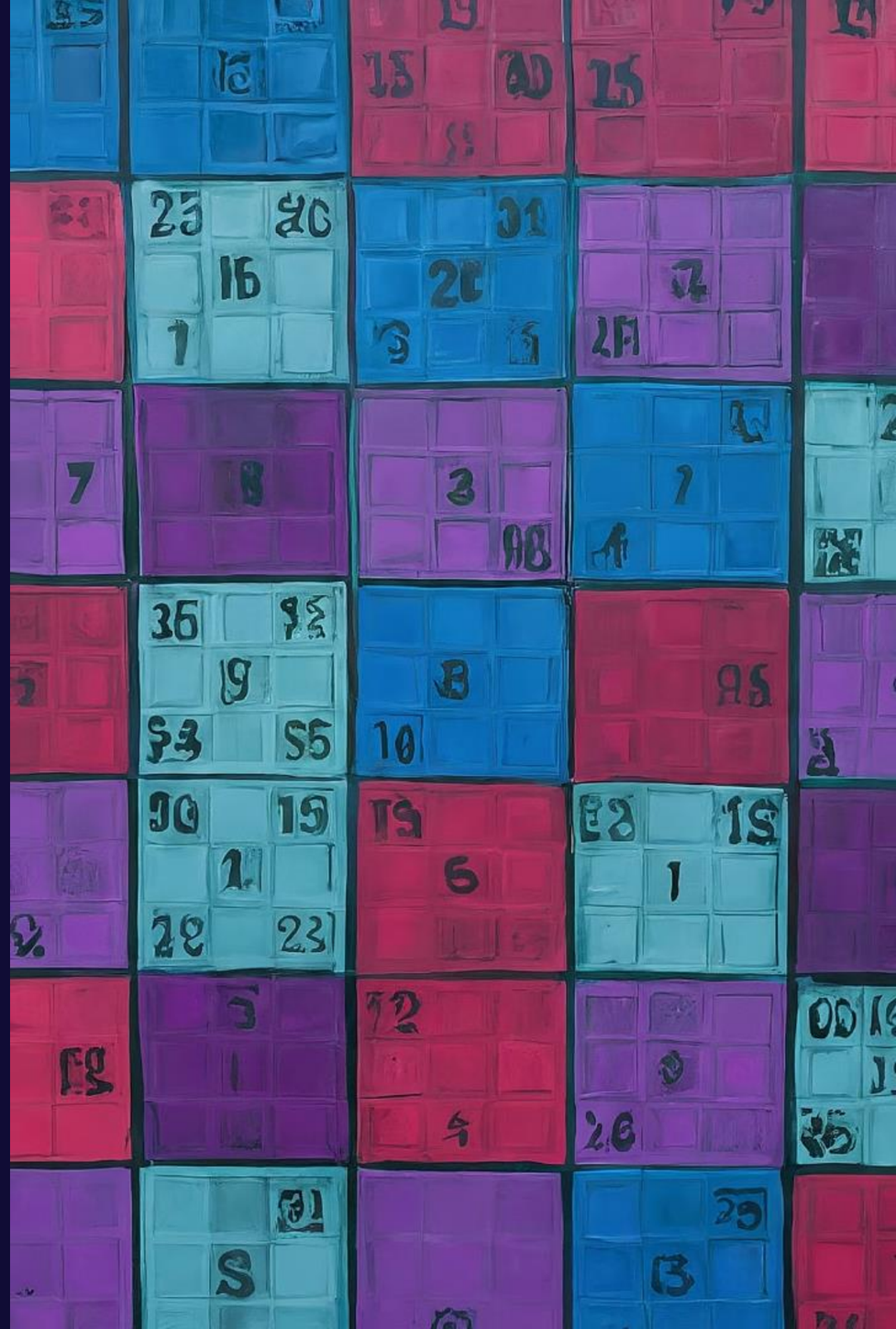
Shiba Narayan Dash(2202040018)

Pranabesh Mishra(2202040019)

SSI Pritam Biswal(2202040020)

Introduction to Sudoku

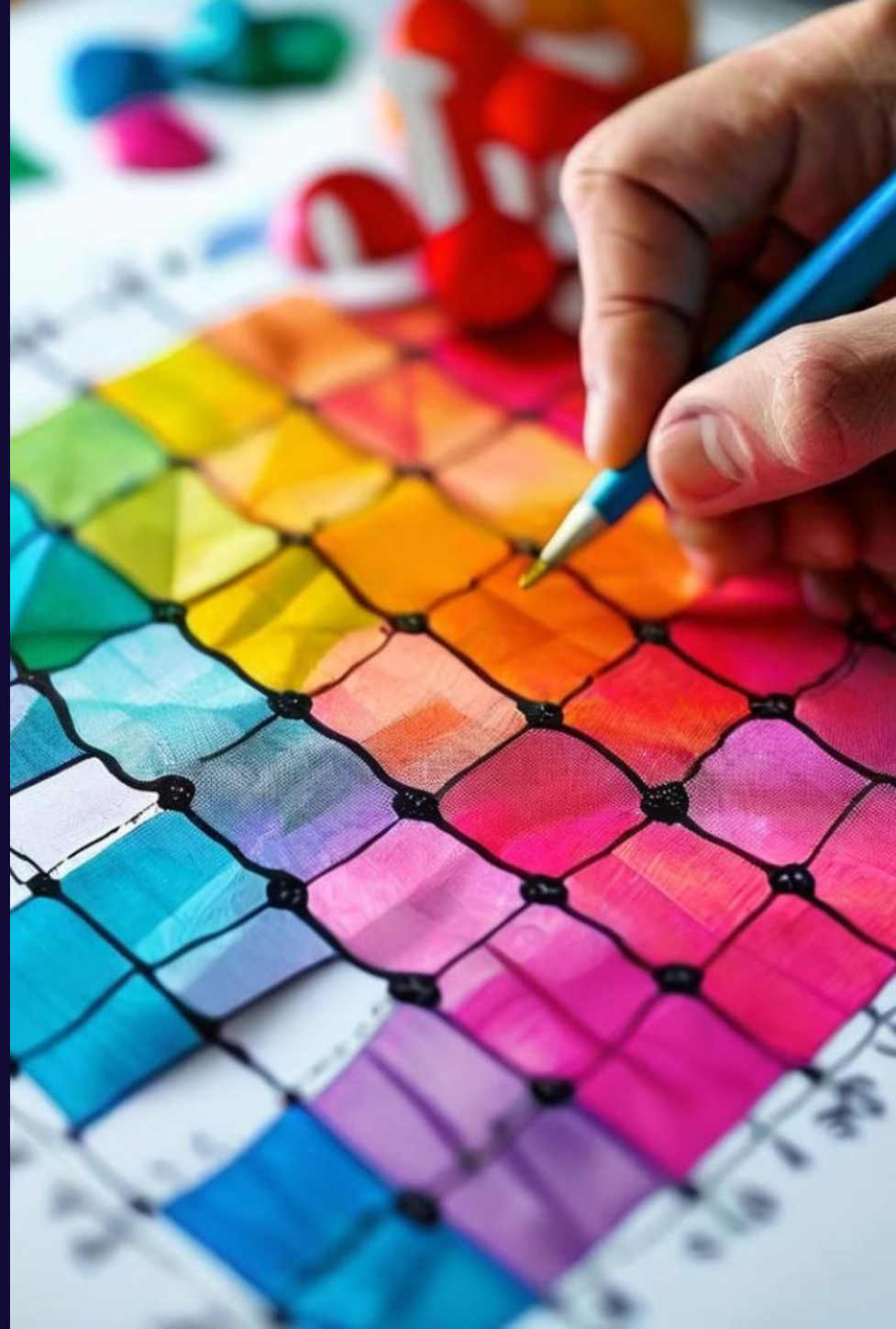
Sudoku is a popular logic-based number placement puzzle that challenges the challenges the mind and exercises critical thinking skills. Players must fill a 9x9 fill a 9x9 grid with digits so that each column, each row, and each of the nine 3x3 the nine 3x3 subgrids contains all of the digits from 1 to 9.



Representing Sudoku as a Graph

To solve Sudoku puzzles using graph coloring algorithms, we first need to need to represent the Sudoku grid as a graph. Each cell in the Sudoku grid grid becomes a node in the graph, and edges are drawn between nodes that nodes that share a row, column, or 3x3 subgrid.

This graph-based representation captures the constraints of the Sudoku problem, allowing us to leverage powerful graph coloring techniques to find a valid assignment of numbers to the empty cells.



Graph Coloring Algorithms

1

Vertex Coloring

Assign colors to the vertices of the graph such that no two adjacent vertices share the same color.

2

Backtracking

A recursive algorithm that tries different color assignments, backtracks when a solution is not possible.

3

Greedy Algorithms

Color vertices one by one, choosing the smallest available color that doesn't conflict with neighbors.

Solving Approach

1

Identify Constraints

Analyze the Sudoku grid to determine the constraints for each cell.

2

Propagate Constraints

Spread the constraints across related cells to eliminate invalid invalid options.

3

Iterative Refinement

Repeatedly apply constraint propagation until until no more deductions can be made.

Constraint propagation is a key technique in solving Sudoku puzzles using a graph-coloring approach. By approach. By identifying the constraints for each cell and propagating them throughout the grid, the solver grid, the solver can eliminate invalid options and narrow down the possible solutions, leading to a more to a more efficient backtracking process.

Time Complexity Analysis

1

Backtracking Algorithm

The time complexity of the backtracking algorithm for solving Sudoku is O(9^(n^2)), where n is the size of the Sudoku grid (usually 9x9).

2

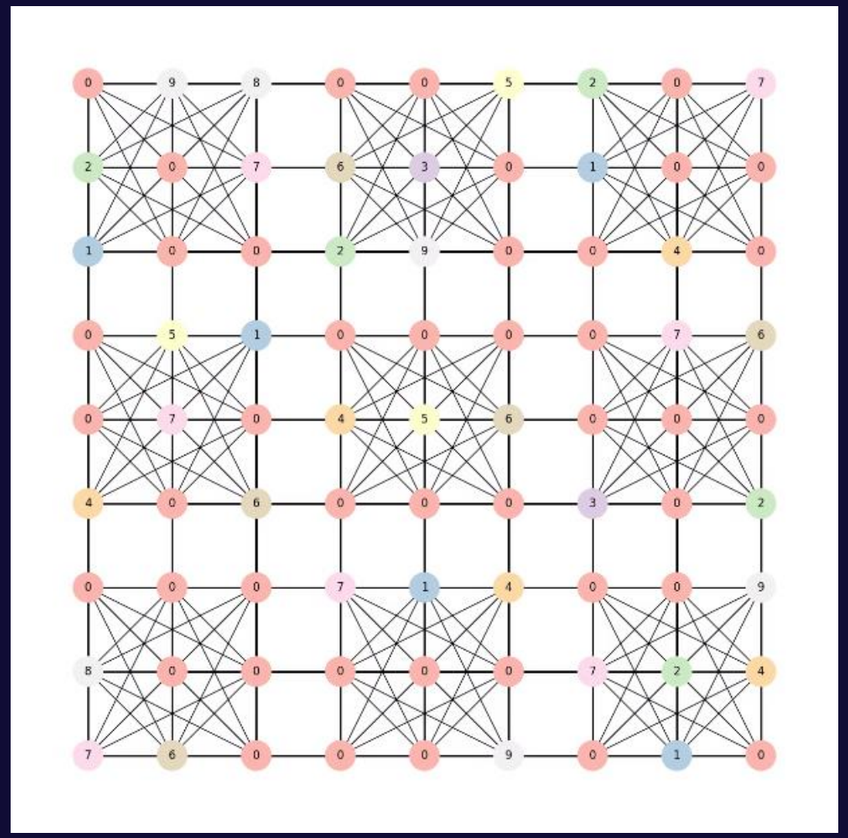
Constraint Propagation

By incorporating constraint propagation techniques, the time complexity can be reduced to O(n^4), significantly improving the solving efficiency.

Implementing the Sudoku Solver

The implementation of the Sudoku solver involves a combination of graph coloring algorithms, backtracking, and constraint propagation techniques. We start by representing representing the Sudoku grid as a graph, where where each cell is a node and the constraints constraints between cells are the edges.

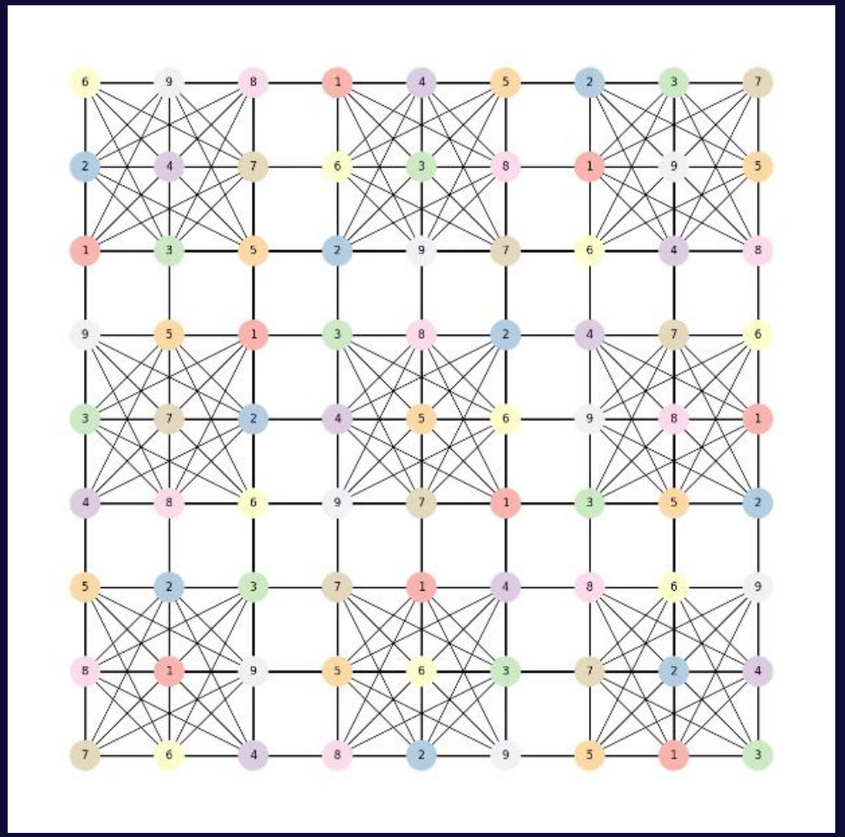
We then apply a graph coloring algorithm to to assign unique values to each cell, ensuring that ensuring that no two adjacent cells have the the same value. This is followed by a backtracking backtracking approach to explore all possible possible solutions and find the correct assignment of values to the grid.



Solving the Sudoku

The Solving of the Sudoku requires backtracking algorithm in which it first iterates over the grid and assigns color and number to the sudoku grid such that no two adjacent vertices have same color.

The backtracking algorithm starts by selecting an selecting an empty cell in the grid and trying out trying out different values one by one. If a value value satisfies the constraints, it is assigned to the to the cell and the algorithm moves on to the the next empty cell. If no value satisfies the constraints, it backtracks to the previous cell and cell and tries a different value. This process continues until all cells are filled, resulting in a in a valid solution to the Sudoku puzzle.



Conclusion

In conclusion, the Sudoku solver using graph coloring has proven to be a versatile and efficient approach. By representing the Sudoku puzzle as a graph and applying graph coloring algorithms, we have demonstrated a robust solution that can handle a wide range of Sudoku puzzles, including the most challenging ones.


TIHANK YOU