**Project Report**

**(Group-2)**

CSE440 (Artificial intelligence)

**Project Title : AI Agent-Based Sudoku Logic Puzzle Solver Using Constraint Satisfaction Techniques**

**Project members:**

1. Md. Shakibur Rahman (ID: 2013534042)
2. Rifa Nanziba Keka (ID: 2021936642)
3. Sabiqun Nahar Ritu (ID: 2021597642)

Section : 01

Date : 8/7/2025

Faculty : MSRB

Title: AI Agent-Based Sudoku Solver Using Constraint Satisfaction Techniques with GUI

1. Introduction

Sudoku is a popular logic puzzle where you fill a 9×9 grid so that each number from 1 to 9 appears exactly once in every row, column, and 3×3 box. Though it looks simple, harder puzzles can be quite challenging and time-consuming to solve by hand.

In this project, we created an AI-based Sudoku solver using Constraint Satisfaction Problem (CSP) techniques. Our goal was to make the AI solve puzzles like a human using smart strategies like backtracking, forward checking, and the Minimum Remaining Value (MRV) heuristic. We also built a simple Tkinter GUI that lets users input puzzles and see the solution instantly, making the tool both practical and easy to use.

2. Objectives

The main goals of this project were:

* To apply AI and CSP concepts to real-world problem solving.
* To implement a backtracking algorithm with forward checking to improve efficiency.
* To incorporate MRV heuristic for smarter decision-making during search.
* To create a simple and intuitive GUI for users to input puzzles and see solutions instantly.

3. Sudoku as a CSP

To model Sudoku as a CSP:

Variables: Each cell in the grid (total 81).

Domains: Values {1–9} for empty cells; single fixed values for pre-filled ones.

Constraints: No repeated digits in any row, column, or 3×3 box.

The AI's job is to assign values to these variables in a way that all constraints are satisfied. This is classic CSP territory and serves as a great example of how AI reasoning applies to games and puzzles.

4. Methodology

Our project has three main components:

AI Agent Logic:

The solver uses recursive backtracking with the MRV heuristic to choose the cell with the fewest options. Forward checking helps prune invalid paths early by reducing related cells' domains after each assignment.

Forward Checking:

When a value is placed, it’s removed from the domains of cells in the same row, column, and subgrid. If any domain becomes empty, the algorithm backtracks.

GUI with Tkinter:

Users can input puzzles into a 9×9 grid. After clicking "Solve", the AI fills in the solution or shows an error if the puzzle is invalid or unsolvable.

5. Tools & Technologies Used

Language: Python, Libraries: Tkinter for GUI, built-in Python structures for logic

Python was chosen for its simplicity, and Tkinter for being easy to use and built into the standard library.

6. Results

The solver works efficiently for most standard Sudoku puzzles and provides real-time results through the GUI. For easier puzzles, the solution is found almost instantly. In case of input errors or unsolvable configurations, the GUI displays an appropriate message.

Sample Output:

Original Puzzle:

[5, 3, 0, 0, 7, 0, 0, 0, 0]

Solved Puzzle:

[5, 3, 4, 6, 7, 8, 9, 1, 2]

7. Advantages

Fast & Efficient: Solves puzzles in under a second.

User-Friendly: Simple GUI for both input and output.

AI in Action: Shows how CSP and AI can solve real problems.

Extendable: Codebase can be adapted for other logic-based games like KenKen or Kakuro.

8. Limitations

Doesn’t handle larger boards (e.g., 16x16) efficiently. Lacks advanced inference methods like AC-3 or arc consistency. No step-by-step visualization of the solving process yet.

9. Future Work

Add a Sudoku puzzle generator with difficulty levels. Include color-coded cells to highlight changes or filled-in values. Support larger grids (16x16, 25x25). Allow exporting solutions to PDF/image. Integrate more AI techniques like AC-3 or even genetic algorithms.

10. Conclusion

This project was a great opportunity to explore how AI concepts like constraint satisfaction and backtracking can be applied to solve everyday problems. Building the solver and GUI helped me understand the core logic behind Sudoku and how to program AI to solve it efficiently. Overall, it’s a solid foundation for future projects in both AI and user-interface development.