Artificial Intelligence
Lab 3 - Sudoku

—

Abstract

This lab involves developing a Sudoku-solving application with a graphical user interface (GUI). The program operates in two modes:

1. **Automated Mode:** Displays the AI agent solving a predefined puzzle.
2. **Interactive Mode:** Allows users to input a custom puzzle for the AI to solve.

The AI uses logic and constraint-solving techniques to ensure accurate solutions, demonstrating the practical application of AI in an interactive environment.

CSP Representation for Sudoku

Sudoku is a 9x9 grid where the goal is to fill the grid such that:

- Each row contains digits 1 to 9 without repetition.
- Each column contains digits 1 to 9 without repetition.
- Each 3x3 sub-grid contains digits 1 to 9 without repetition.

CSP Definition:

1. **Variables**:
   Each cell in the grid is a variable Xij, where i is the row index and j is the column index.
2. **Domains:**
   The domain of each variable is P{1 , 2 , 3, …… , 9 } (initially). For pre-filled cells, the domain contains only the given number.
3. **Constraints:**
   - Row Constraint: No two cells in the same row can have the same value.
   - Column Constraint: No two cells in the same column can have the same value.
   - Sub-grid Constraint: No two cells in the same 3x3 sub-grid can have the same value.

Backtracking in Sudoku

Backtracking is a search-based algorithm to solve Sudoku by systematically assigning values to variables (cells) and checking constraints.

Steps:

1. Start with an empty or partially filled grid.
2. Pick an unassigned cell (variable).
3. Assign a value from the cell's domain.

4. Check if the assignment satisfies all constraints:
    - If consistent: Move to the next unassigned cell.
    - If inconsistent: Backtrack, undo the assignment, and try a different value.
5. Repeat until the grid is completely and validly filled.

AC-3 in Sudoku

The AC-3 algorithm can preprocess the Sudoku grid by reducing the domains of variables before solving with backtracking.

How AC-3 Works in Sudoku:

1. Arcs:
   Each pair of related cells (same row, column, or sub-grid) represents an arc (Xij - Xnm).
2. Making Arcs Consistent:
    - For each arc we check if Xij has for each corresponding value in domain a value in the Xnm domain without any contradiction . if yes go ahead , else remove this value from Xij and add all arc which were towards Xij again in queue

Advantages for Sudoku:

- Finds a valid solution if it exists.
- Eliminates impossible values early, reducing the number of choices for each cell.
- Speeds up backtracking by working with smaller domains.

Disadvantages:

- Inefficient for complex grids due to the combinatorial explosion of possibilities.

Data structures used :

- List
- Dictionary
- Queue

Complexity:

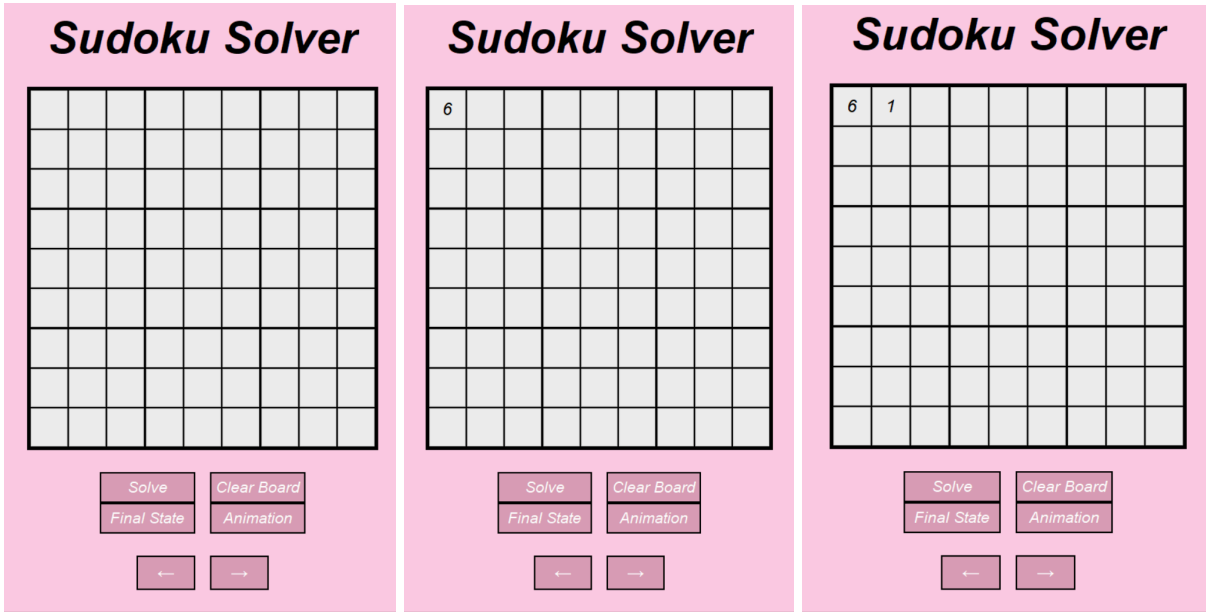
- Backtracking : like dfs O(d^n) as d =: # of values / n =: # of variables.
- Arc consistency : O(n^2 * d^3)

Note:

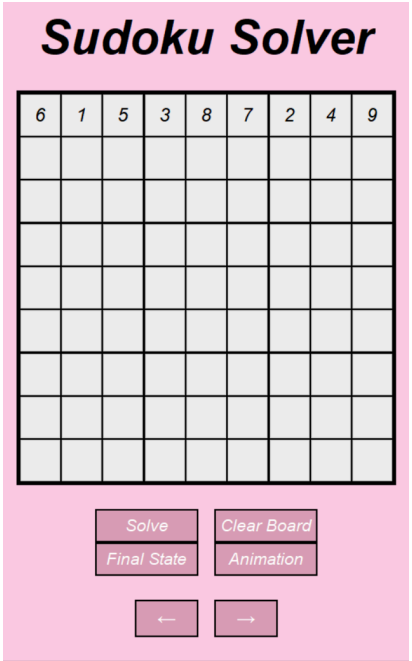
- We use to choose variables to be assigned as variables which have less number of values of domain.
- If there are many variables with only one value in the domain then assign them all at once.
- When we choose the value assigned to variable we make it random to make sure that for empty board there is many multiple boards rather than one.

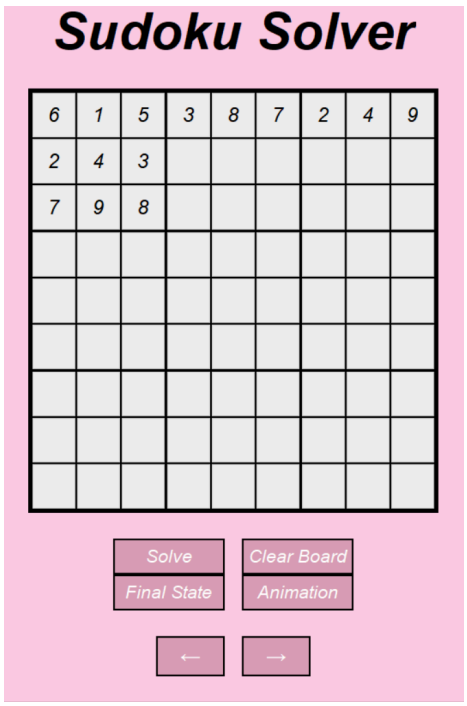
Sample Runs Mode 1:

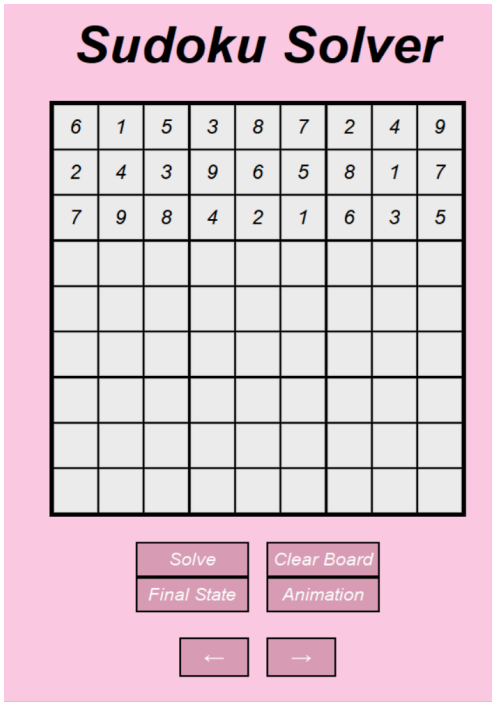
1- Initial State: Empty Board

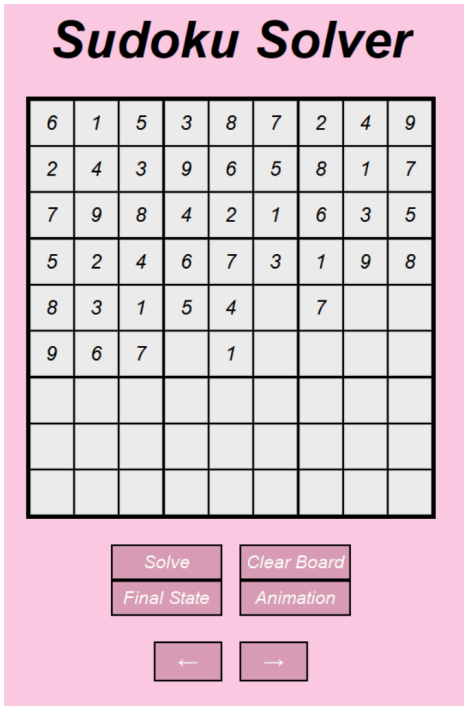


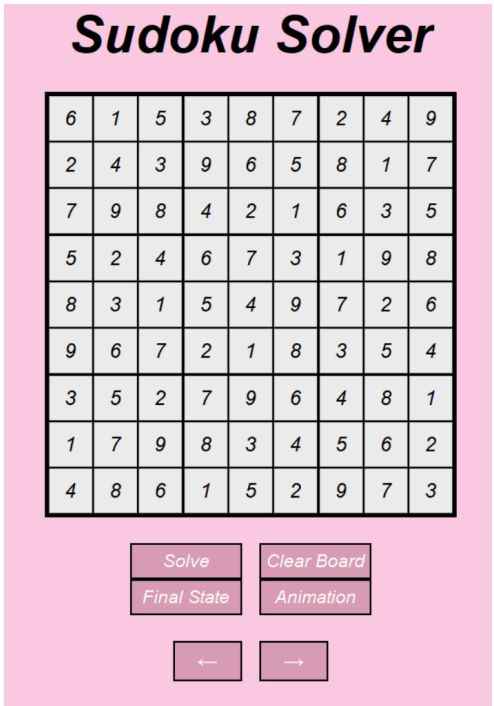
– The model keeps filling one cell in the first row until we reach this state, then we fill each cell in the first subgrid until we reach this state




Sudoku Solver
6 1 5 3 8 7 2 4 9
2 4 3
7 9 8
Solve
Final State
Clear Board
Animation
←
→


Sudoku Solver
6 1 5 3 8 7 2 4 9
2 4 3 9 6 5 8 1 7
7 9 8 4 2 1 6 3 5
Solve
Final State
Clear Board
Animation
←
→


Sudoku Solver
6 1 5 3 8 7 2 4 9
2 4 3 9 6 5 8 1 7
7 9 8 4 2 1 6 3 5
5 2 4 6 7 3 1 9 8
8 3 1 5 4 7
9 6 7 1
Solve
Final State
Clear Board
Animation
←
→


Sudoku Solver
6 1 5 3 8 7 2 4 9
2 4 3 9 6 5 8 1 7
7 9 8 4 2 1 6 3 5
5 2 4 6 7 3 1 9 8
8 3 1 5 4 9 7 2 6
9 6 7 2 1 8 3 5 4
3 5 2 7 9 6 4 8 1
1 7 9 8 3 4 5 6 2
4 8 6 1 5 2 9 7 3
Solve
Final State
Clear Board
Animation
←
→

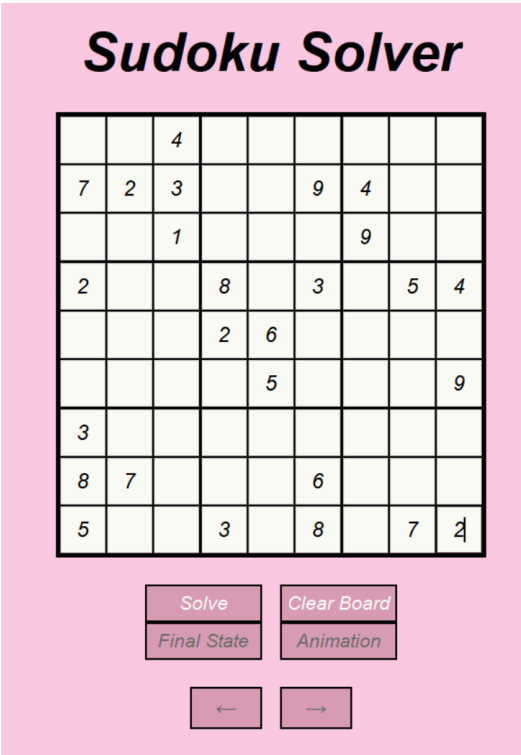
Last step of updating domains to reach the final solution – Time Taken = 0.45 Sec

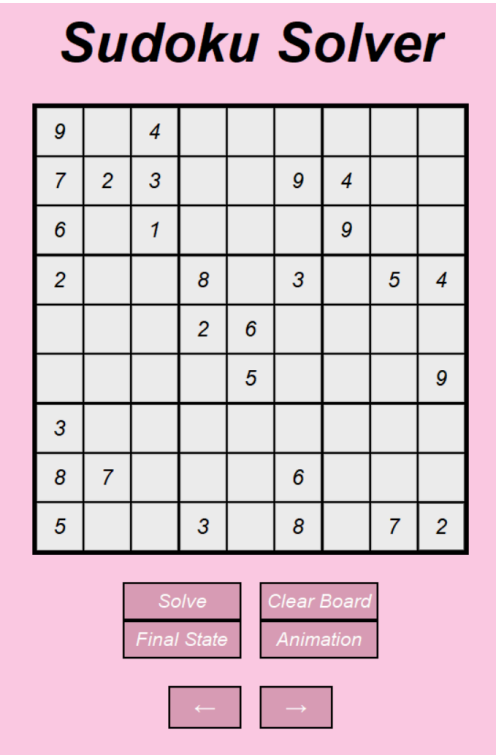
```
this is i = V95, this is j = V97
for V95 this is domain [3, 4]
for V97 this is domain [4]
for V95 this is domain after Removal of 4 : [3]
this is i = V92, this is j = V82
for V92 this is domain [3, 9]
for V82 this is domain [3]
for V92 this is domain after Removal of 3 : [9]
this is i = V85, this is j = V82
for V85 this is domain [3, 4]
for V82 this is domain [3]
for V85 this is domain after Removal of 3 : [4]
this is i = V86, this is j = V89
for V86 this is domain [2, 5]
for V89 this is domain [5]
for V86 this is domain after Removal of 5 : [2]
this is i = V96, this is j = V93
for V96 this is domain [2, 5]
for V93 this is domain [2]
for V96 this is domain after Removal of 2 : [5]
this is i = V99, this is j = V96
for V99 this is domain [1, 5]
for V96 this is domain [5]
for V99 this is domain after Removal of 5 : [1]
Time taken: 0.45 seconds
```

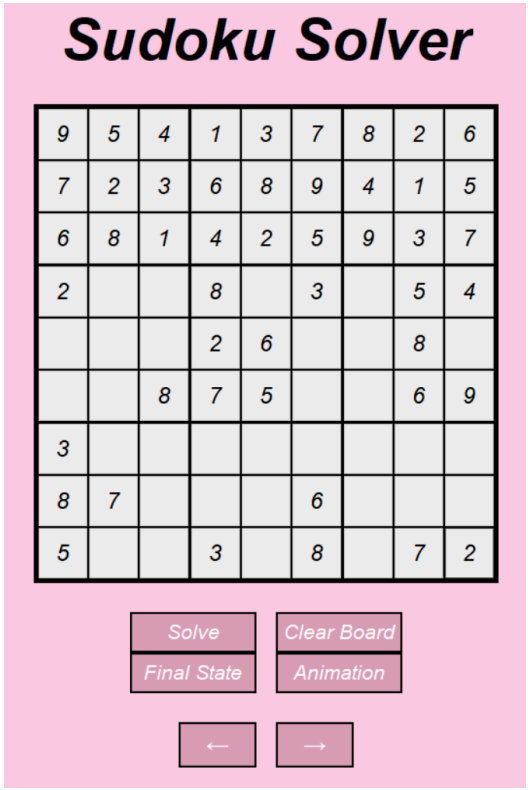
2- Initial state: New York Times [Daily Hard Challenge]

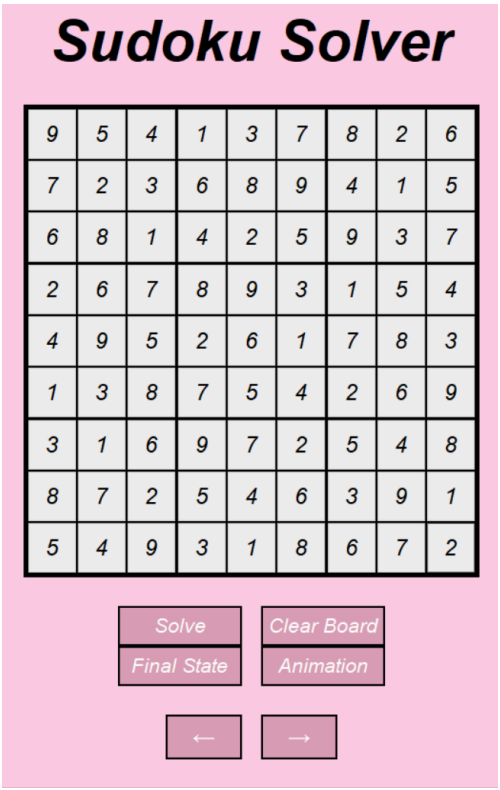
First step of updating domains

```
this is i = V11, this is j = V91
for V11 this is domain [1, 2, 3, 4, 5, 6, 7, 8, 9]
for V91 this is domain [5]
for V11 this is domain after Removal of 5 : [1, 2, 3, 4, 6, 7, 8, 9]
this is i = V11, this is j = V81
for V11 this is domain [1, 2, 3, 4, 6, 7, 8, 9]
for V81 this is domain [8]
for V11 this is domain after Removal of 8 : [1, 2, 3, 4, 6, 7, 9]
this is i = V11, this is j = V21
for V11 this is domain [1, 2, 3, 4, 6, 7, 9]
for V21 this is domain [7]
for V11 this is domain after Removal of 7 : [1, 2, 3, 4, 6, 9]
this is i = V11, this is j = V33
for V11 this is domain [1, 2, 3, 4, 6, 9]
for V33 this is domain [1]
for V11 this is domain after Removal of 1 : [2, 3, 4, 6, 9]
this is i = V11, this is j = V41
for V11 this is domain [2, 3, 4, 6, 9]
for V41 this is domain [2]
for V11 this is domain after Removal of 2 : [3, 4, 6, 9]
```



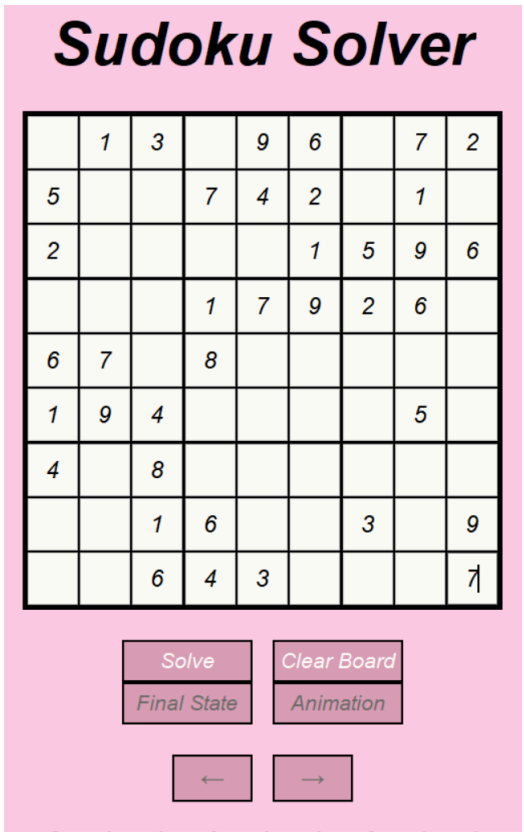


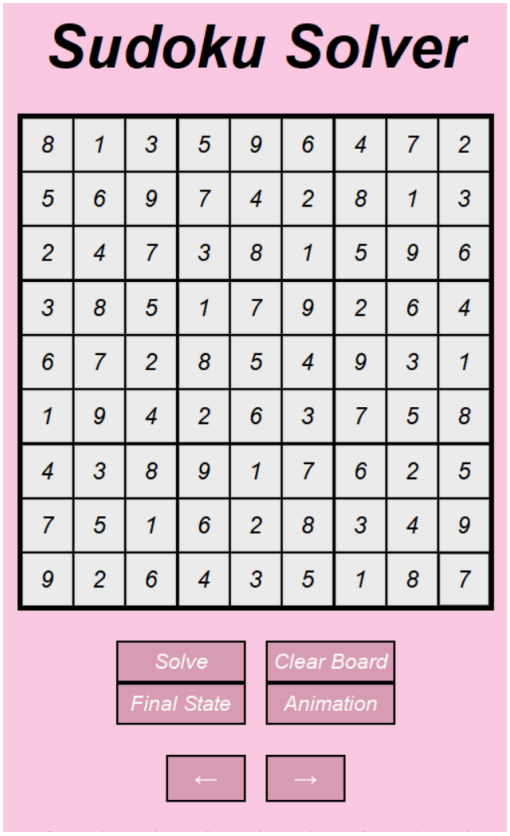




– Time taken = 0.16 Sec

3- Initial state: New York Times [Daily Easy Challenge]






```
for V88 this is domain after Removal of 8 : [4]
this is i = V59, this is j = V56
for V59 this is domain [1, 4]
for V56 this is domain [4]
for V59 this is domain after Removal of 4 : [1]
this is i = V58, this is j = V56
for V58 this is domain [3, 4]
for V56 this is domain [4]
for V58 this is domain after Removal of 4 : [3]
this is i = V92, this is j = V82
for V92 this is domain [2, 5]
for V82 this is domain [5]
for V92 this is domain after Removal of 5 : [2]
this is i = V86, this is j = V82
for V86 this is domain [5, 8]
for V82 this is domain [5]
for V86 this is domain after Removal of 5 : [8]
this is i = V72, this is j = V82
for V72 this is domain [3, 5]
for V82 this is domain [5]
for V72 this is domain after Removal of 5 : [3]
this is i = V76, this is j = V79
for V76 this is domain [5, 7]
for V79 this is domain [5]
for V76 this is domain after Removal of 5 : [7]
Time taken: 0.23 seconds
```

→ Solved in only one step as the domain of each cell has one value only

And the domain reduction lead to make all the empty cells have a domain with size 1

– Time Taken = 0.23 Seconds

Player Mode Implementation:

The game starts with an empty board, which is solved to produce a valid final state that meets the rules of the puzzle. This final state serves as the completed solution for the game. To generate a playable puzzle, a specific number of slots are randomly cleared (made empty) based on the chosen difficulty level. The number of empty slots increases with the difficulty, creating progressively more challenging puzzles for the player to solve.

Here are the details of how difficulty levels affect the number of empty slots:

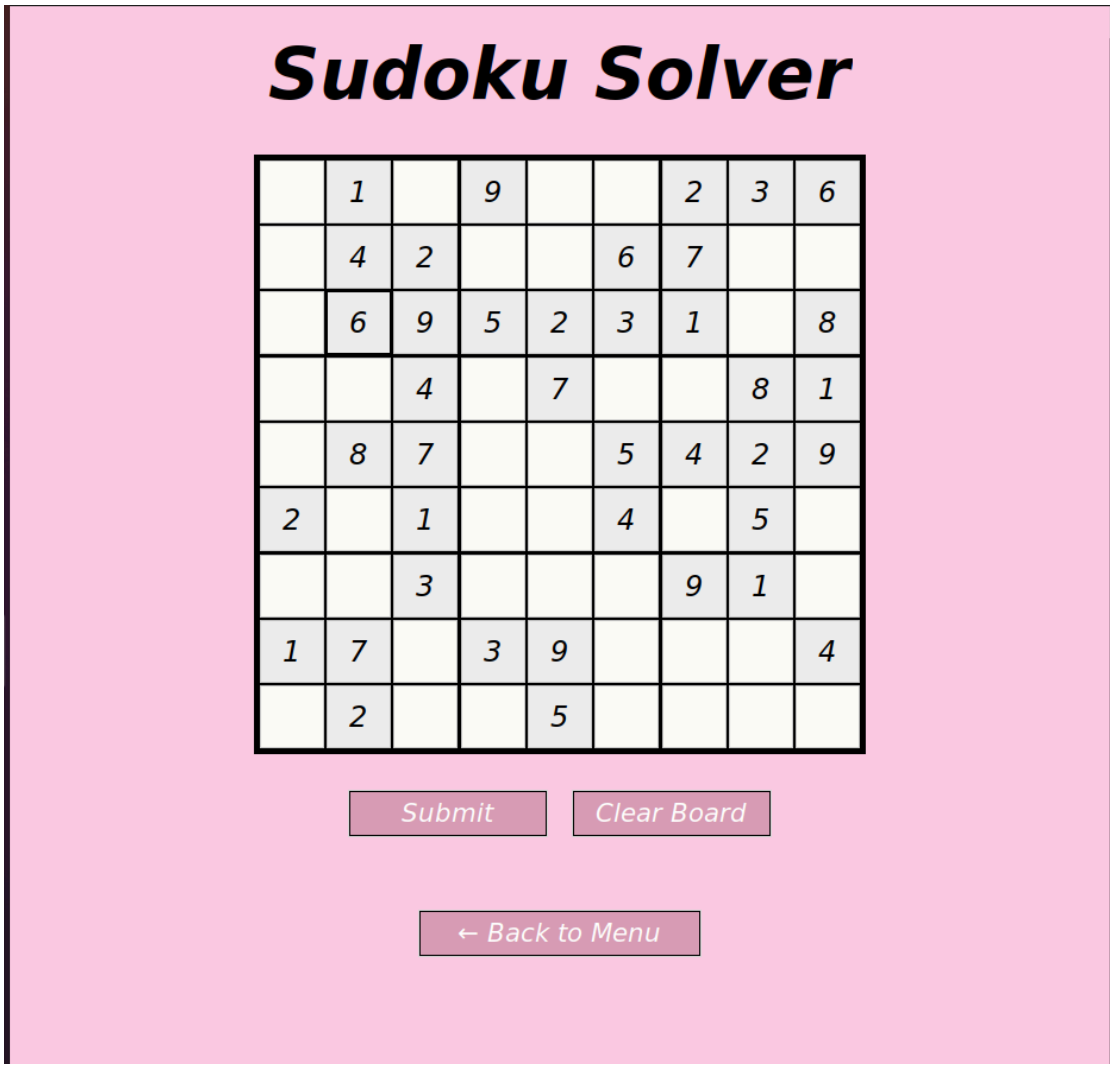
- **Easy:** 41 slots are randomized and left empty, resulting in a relatively simple puzzle to solve.
- **Medium:** 51 slots are randomized and left empty, providing a moderate level of challenge.
- **Hard:** 58 slots are randomized and left empty, creating a highly challenging puzzle suitable for advanced players.

This approach ensures that each difficulty level provides a unique and appropriate challenge while maintaining the solvability of the puzzle. The randomization ensures variety, so each game feels fresh and engaging for players.

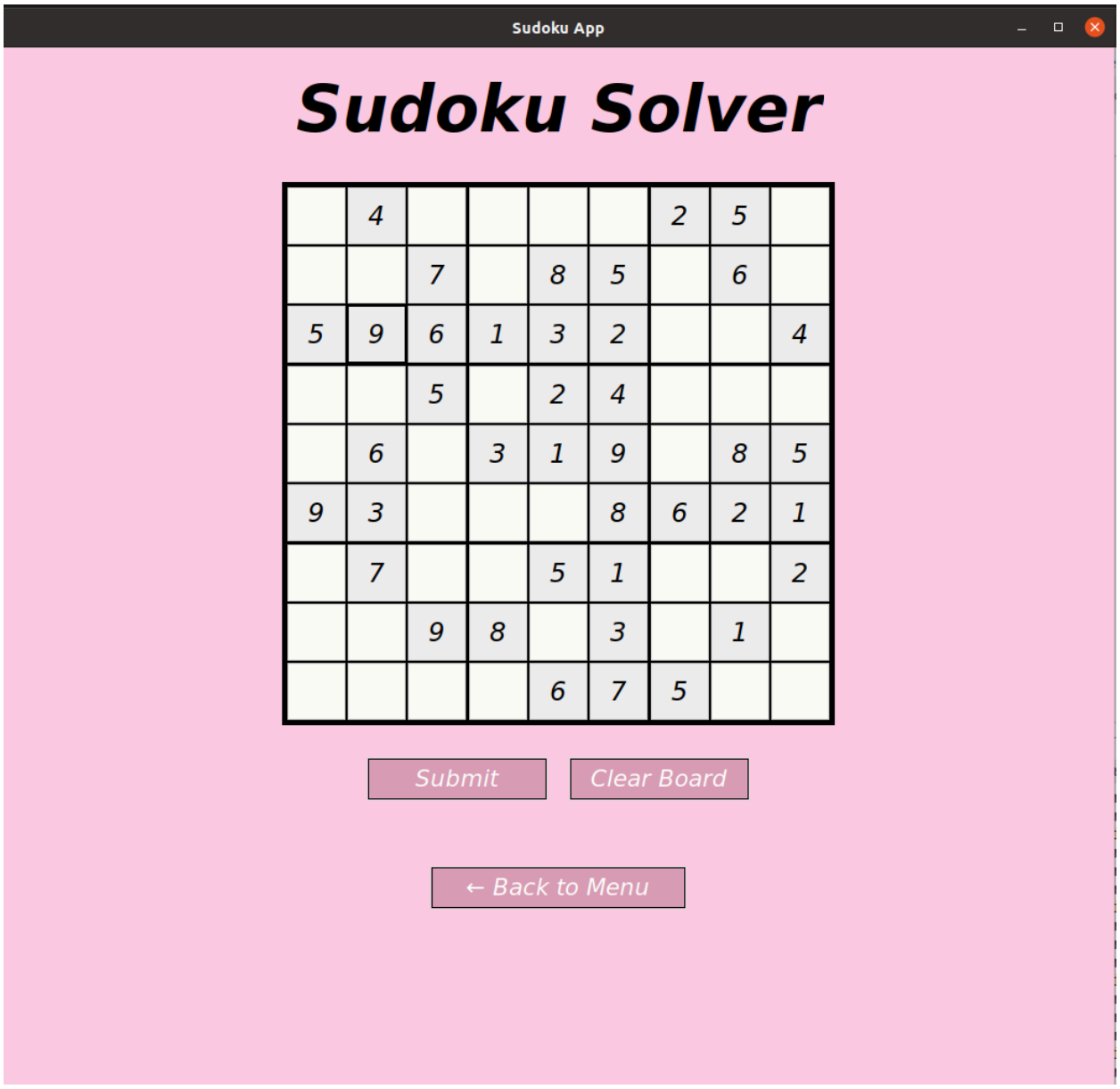
Additional Sample runs: Mode: 2

Easy level:

1- Time taken: 0.60 seconds

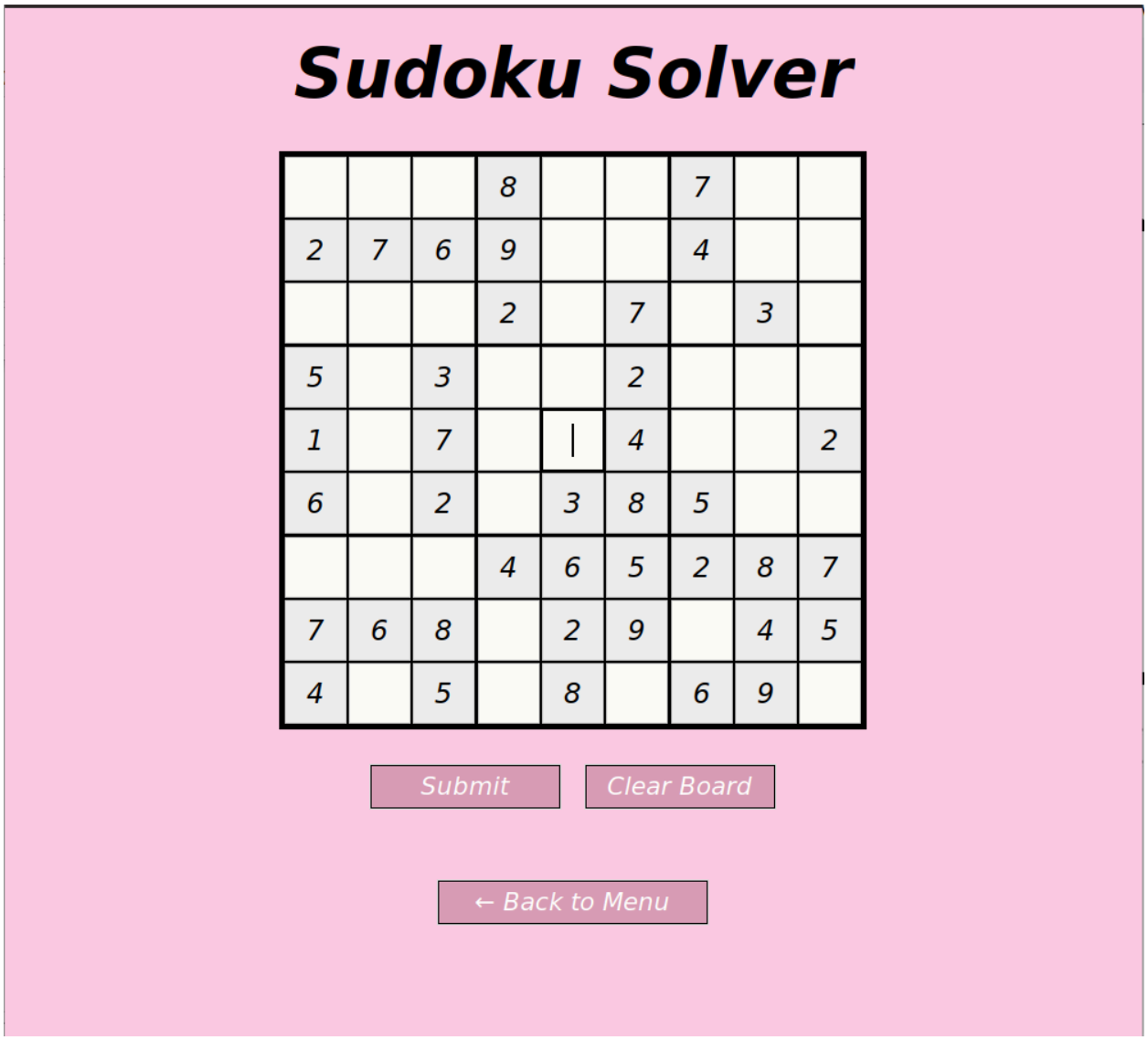


2- Time taken: 0.64 seconds



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

3- Time taken: 0.56 seconds



Solving the Puzzle:

Since the puzzle guarantees a unique final solution, players are given two modes of interaction for solving it:

1. **Submit the Entire Board at Once:**
    - The player has the option to fill in all the empty slots and submit the completed board for validation in one step.
    - This approach is suited for players who prefer to solve the puzzle entirely before confirming their solution.
2. **Play by Play (Step-by-Step):**
    - The player can fill in slots incrementally, submitting each move individually as they progress.
    - This allows for real-time feedback, enabling the player to adjust their approach if a mistake is identified along the way.

These flexible solving options cater to different playstyles, allowing users to choose between a more methodical, step-by-step experience or a full-on challenge of solving the board in one go. The uniqueness of the final solution ensures fairness and clarity in validating the player's submissions.

Sudoku Solver

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

WOW!!

Correct! Keep going!

OK

Submit Clear Board

← Back to Menu

Sudoku Solver

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

Congratulations !!

Congratulations

OK

Submit Clear Board

← Back to Menu

Sudoku Solver

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

Puzzle Solving Failed

Oops! Try a different Solution

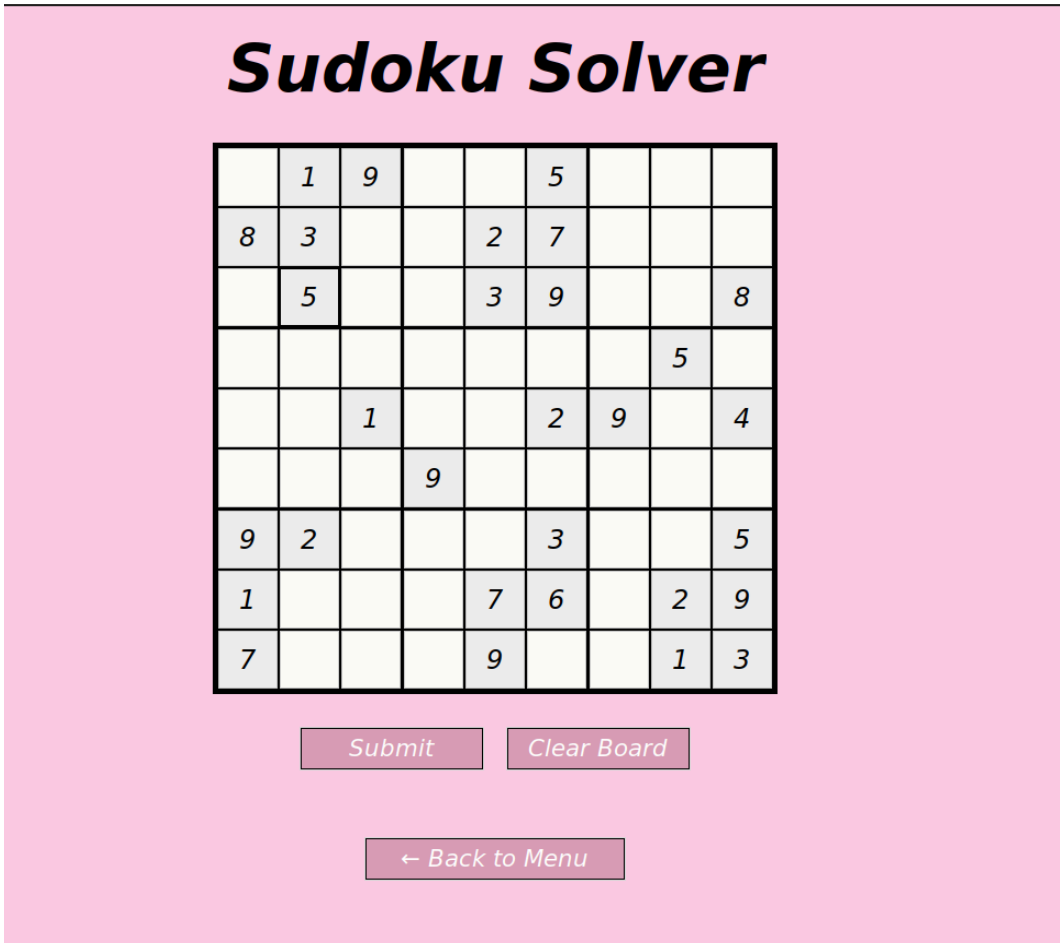
OK

Submit Clear Board

← Back to Menu

Medium Level:

Time taken: 0.58 seconds



Hard Level:

Time taken: 0.59 seconds