Sudoku Solvers Milestone 2

Alice Luce, Adrian Rodriguez, Michael Richards, Jaden Krekow

Research Results

- We have over a dozen candidate algorithms to explore and compare as we progress into the project
- Despite many algorithms and techniques being explored, nearly every paper operated on Sudoku as a 2D array, 2D linked structure, or 1D list of digits. The 2D array is by far the most common and treated as the most efficient in the general case

Brute Force Implementation

- A brute force algorithm has been implemented for testing as we create the GUI and preparing the application itself.
- It takes several seconds to solve most boards, and several minutes to generate boards via random placement of hints and testing for multiple solutions.

Current state of GUI

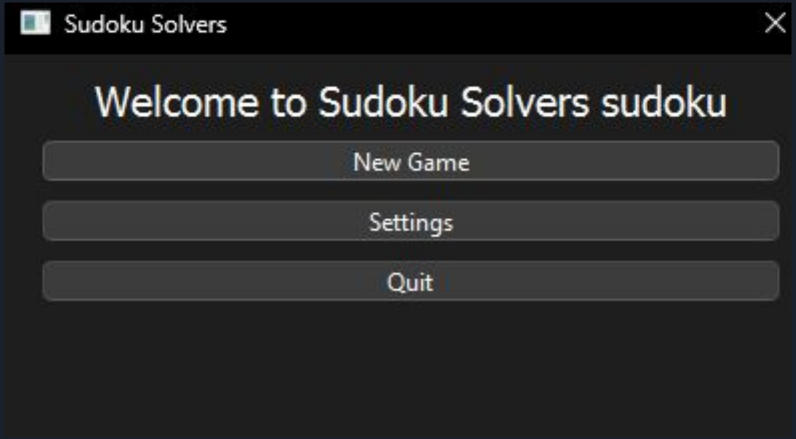
- Create a GUI that can implement basic user interface for both playability and learning
- Basic features implemented such as generation, solving, timers, and sudoku board
- Understand limitations of PySide 6
- Will be useful in testing databases, algorithms, and showing the user educational information both on Sudoku and generation algorithms we used to generate a board
- Current focusing on implementation of brute force algorithm and results shown

Future Additions to GUI

- Adding the choice for users to pick a specific algorithm to use for their board generation
- Adding algorithm information and rules to the user interface that lets users learn while playing
- Enhance existing features through code refactoring or visual changes in the GUI to enhance user experience
- Implement option to use stored boards from Database to assist in board generation time

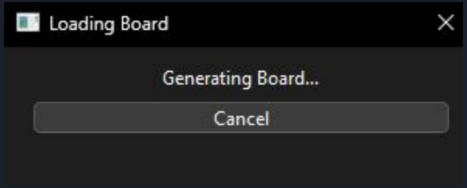
GUI Menu and Multithreading

- Application starts off with the menu
  - Start game, Settings, Quit
- Start game -> loading screen while board is generated on a separate thread-> board is displayed once done



Sudoku Solvers

Welcome to Sudoku Solvers sudoku

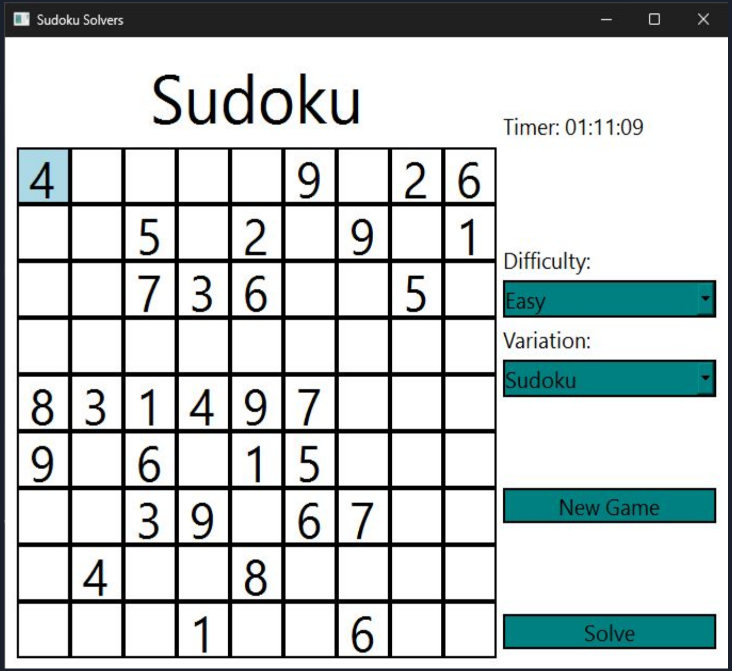


Loading Board

Generating Board...

GUI

- Two instances of the UI with easy and normal difficulty
- Using Brute Force

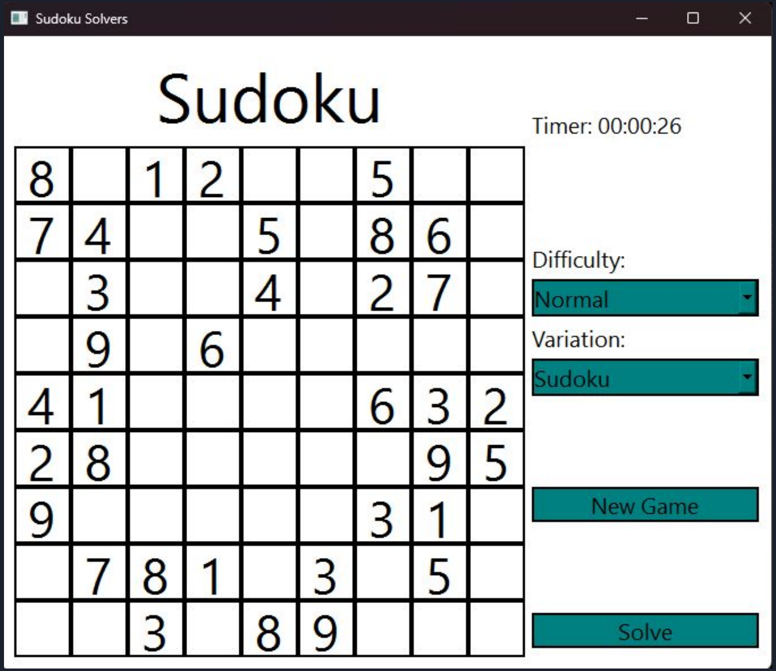


Timer: 01:11:09

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

Difficulty: Easy

Variation: Sudoku



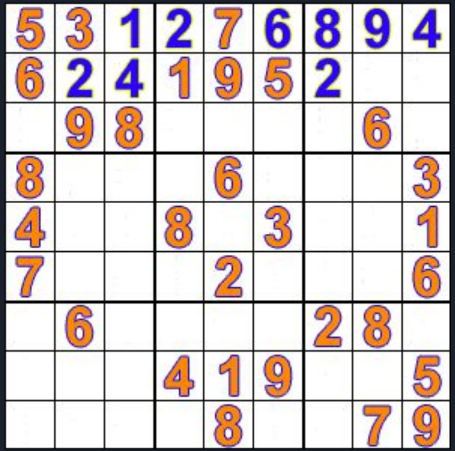
Timer: 00:00:26

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

Difficulty: Normal

Variation: Sudoku

Database



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

- SQLite3 Database Created
- Board
  - Puzzle ID: Unique identifier for puzzle
  - Raw Board: Initial unsolved Sudoku Puzzle
  - Solved Board: Final solved Sudoku Puzzle
  - Board State: Contains current board data filled in from user (to save history)
- “Snapshots” will be taken of the board state throughout generation process
  - Will be used to visualize auto-solver algorithm to the user
- Snapshot
  - Snapshot ID: Integer identifier for snapshot that increments by one
  - Puzzle ID: Unique identifier for puzzle Snapshot is linked to
  - Board State: Contains board data at time of snapshot

	Completion	Alice	Adrian	Jaden	Michael	To Do
Sudoku Implementation		50%	50%			Enable interaction between algorithms and GUI
Sudoku testing		50%	50%			Ensure visual display represents accurate data (GUI updates cells)
Data structure and solution research	100%	100%				Develop a novel way to store and access Sudoku boards?
Snapshot/Board Database				100%		
GUI development			50%		50%	Implement GUI and enable functional use of buttons and other features
Tool setup and control					100%	

	Alice	Adrian	Jaden	Michael
Efficiently Solve Sudoku boards	Implement and compare several algorithms to find the most time efficient ones			Help create implementations of algorithms as well as analyze computation time.
Efficiently generate and store well-formed Sudoku boards				
Develop efficient ways to determine the solvability of a Sudoku board	Attempt to develop a novel solution or at least minimize time with current solutions			
Create a functional GUi		Create a GUI that will be used to navigate features used to play Sudoku and receive user input		Add functionality to display the step by step process that the solving algorithm is taking to solve the puzzle.
History functionality for puzzles			Add functionality into GUI for saving puzzles, viewing history of puzzles, and loading previous puzzles	