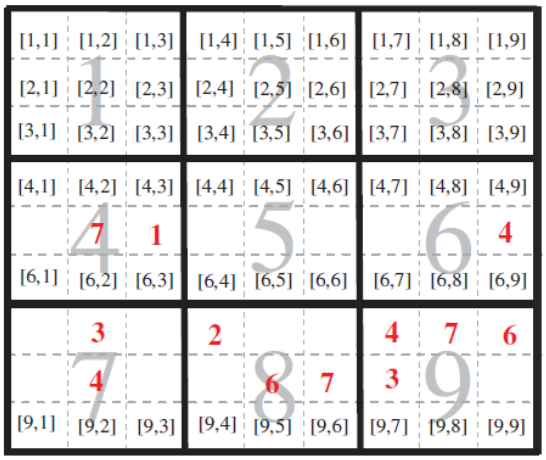
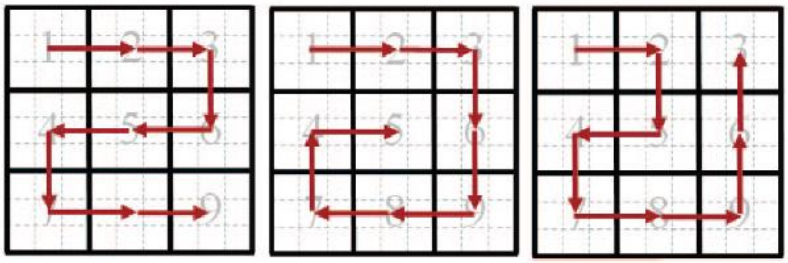
Research Topic: Sudoku Solver

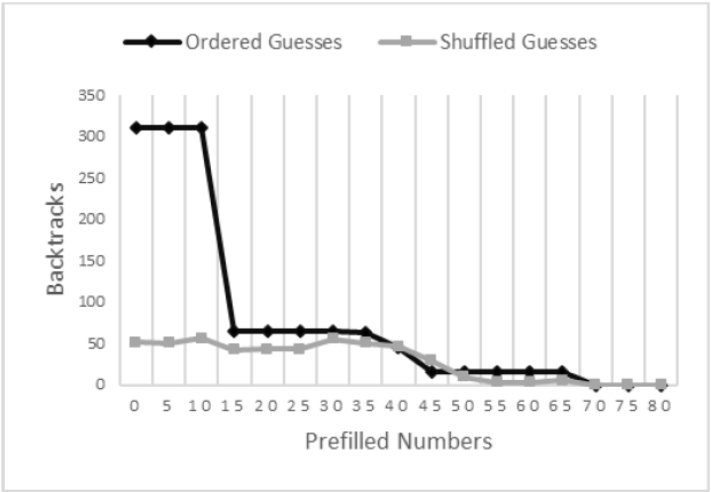
Objective: Be able to solve a 9x9 sudoku puzzle in an efficient manner.

Sample data:

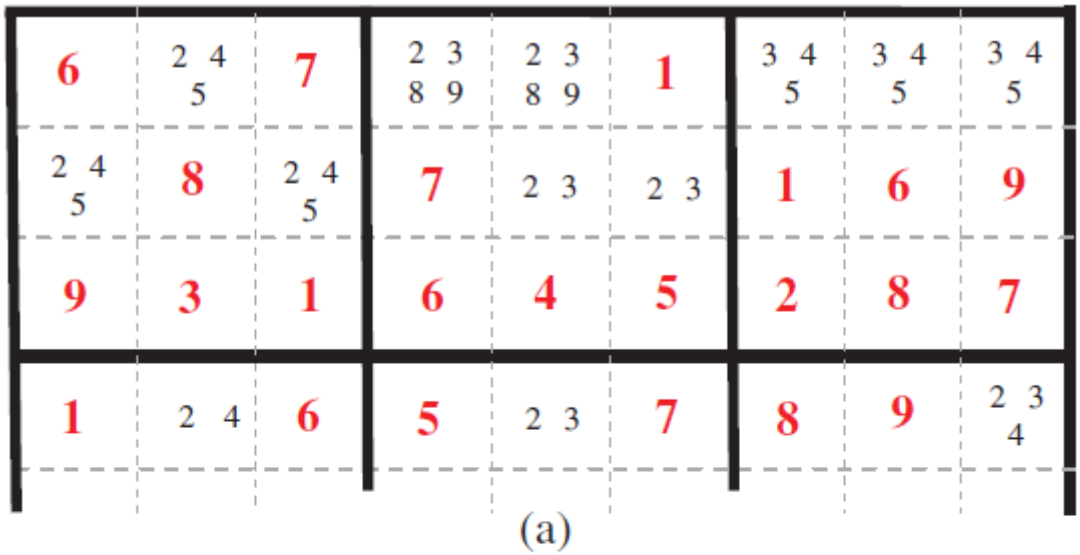


Sample Approaches:





Sample Situations:



(a)

(b)

Datasets:

8400 sudoku puzzles and solutions: https://mypuzzle.org/sudoku

1 million sudoku puzzles and solutions: https://www.kaggle.com/datasets/bryanpark/sudoku

9 million sudoku puzzles and solutions: https://www.kaggle.com/datasets/rohanrao/sudoku

Techniques:

Brute Force (Naïve) with backtracking

Rule based with backtracking

Ant Colony Genetic Algorithm

Tree-based

Neural Network

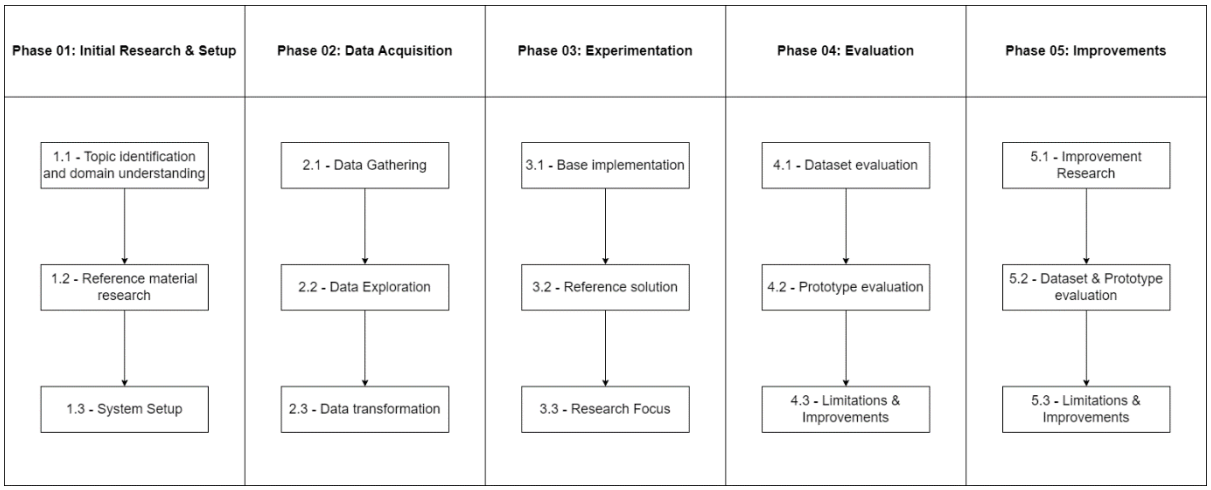
Sample solutions

https://www.askpython.com/python/examples/sudoku-solver-in-python

https://www.techwithtim.net/tutorials/python-programming/sudoku-solver-backtracking/

https://medium.com/@ev.zafeiratos/sudoku-solver-with-python-a-methodical-approach-for-algorithm-optimization-part-1-b2c99887167f

Suggested Approach:



1 Initial Research & setup

1.1 Understand the rules of the game are, what data is available and what approaches exist.

1.2 Find academic literature about topic, find dataset(s), find reference material for techniques (code, tutorial, documentation).

1.3 Setup GIT, Python, Anaconda, Python Virtual Environment (optional), IDE (VSCode).

2 Data acquisition

2.1 Download dataset or scrape website, understand structure.

2.2 Create Jupyter notebook to load dataset. Explore it, identifying nulls, target variable, different target classes, explore correlation, explore distribution (is target variable balanced/unbalanced).

2.3 Prepare data for processing (convert from string to 2-dimensional array, create validation and completion check algorithms).

3 Experimentation

3.1 Create a simple implementation, even heavily based on 3rd party, naïve brute force with backtracking, searching by row and guessing in a sequential manner is a good start.

3.2 Identify another research and determine what they did differently. Add different searching and guessing algorithms. Revise your code to approach 3rd party research so you can compare.

3.3 The research focus is to be able to complete the puzzle in a reasonable time with the least backtracks.

4 Evaluation

4.1 Compare your dataset and the changes you made with 3rd party research.

4.2 Compare the outcome of your technique with that of other parties (it does not need to be better but be able to determine how yours compares).

4.3 Reflect and identify areas of improvements.

5 Improvements

5.1 Consider rule based and other approaches.

5.2 Consider pushing to an API.

5.3 Consider generating your own puzzles.

5.4 Consider adding tracking of solutions.

5.5 Consider code optimization or GPU use (NVIDIA CUDA).

5.6 Calculate the time complexity to solve problems using your algorithms (big O notation).