Playing Games and Solving Puzzles using AI

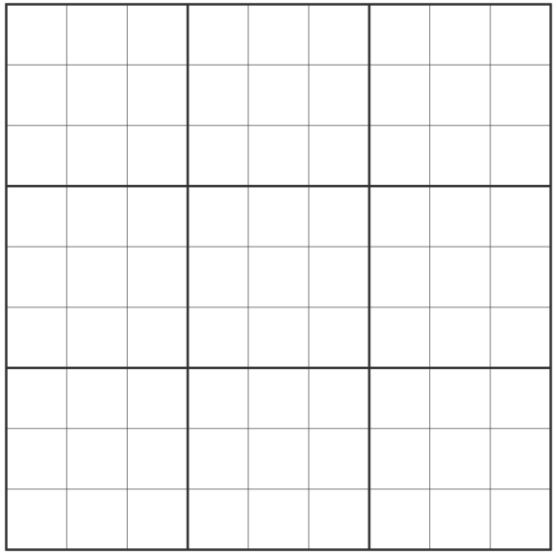
Dimitar Seraffimov

A presentation submitted in part fulfilment of the degree of

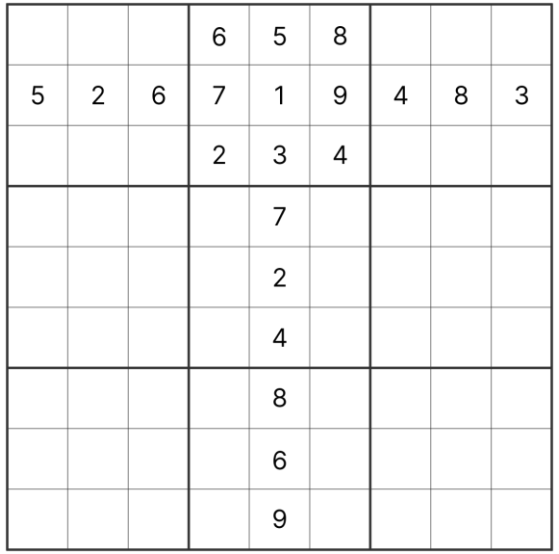
BSc in Computer Science with Software Engineering

Supervisor: Magnus Wahlstrom

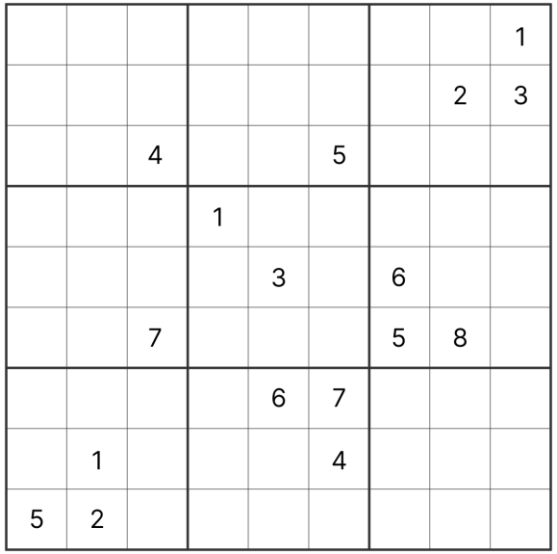
Sudoku Mobile App



An empty Sudoku grid



Partially completed grid



Puzzle with 17 clues
(extreme level)

Aims & Objectives

Harmonie in Design & Software:
balance between seamless user experience and professionally implemented back-end software

Minimalist Design Philosophy:
minimalist yet effective principles and processes in the application's design

Sudoku Puzzle Generation and Difficulty Levels:
generating unique Sudoku puzzles with various difficulty levels, ensuring endless game variants

Difficulty Level	Empty Cells
1 (Easy)	40 to 45
2 (Medium)	46 to 49
3 (Hard)	50 to 53
4 (Extreme)	54 to 58

Empty Cells for each Difficulty Level [1]

Enhanced In-Game Functionalities:

key functionalities like puzzle validation, hints, step-back option, and time tracking to boost user engagement and competitive play

Future Extension to Machine Vision:

plan to explore integration of machine vision for inputting new puzzles - aiming to add advanced user interactions and technological complexity

Theoretical Analysis

N-Queens Problem, 1st PoC Program

- classic problem, placing N number of queens on an N×N chessboard without mutual threats
- illustrates the implementation of recursive backtracking
- demonstrates algorithm's ability to explore all possible configurations

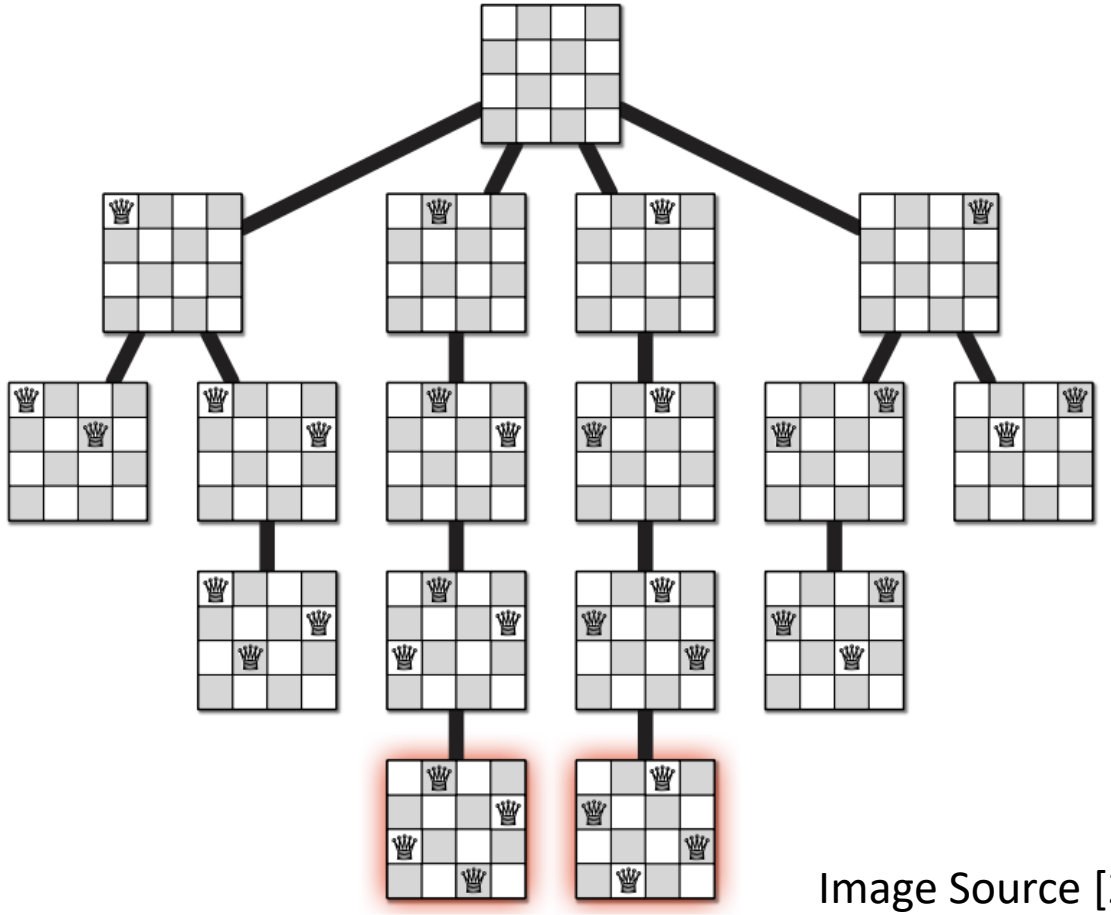


Image Source [2]

Theoretical Analysis

Backtracking Algorithm - 3rd PoC Program

Implementing **Backtracking algorithm** is the key to exhaustive search programs correctly and efficiently.

Explained in simple terms, backtracking is a special approach to problem-solving that involves:

- depth-first search on an implicit graph of configurations;
- exploring all possible solutions to a problem;
- eliminating those solutions that fail to satisfy the constraints;
- "backtracking" to a previous step and trying alternative solutions.

Theoretical Analysis

User Interface Theory - 2nd PoC Program

UML Diagrams:

utilizes UML diagrams for a structured visual representation of the system's architecture

User Stories:

help maintain a human-centric approach, aligning the development with end-user preferences

Agile Methodology Influence:

Agile methodologies [3] influenced the GUI design process by keeping development concise

Practical Application of Knowledge:

applied methods, learned in "CS2800: Software Engineering" to the project's core development

Future Development

Automated Puzzle Generation:

creating fully completed Sudoku puzzles and then removing cells to form the puzzle grid

classify puzzles into four difficulty levels:

Easy, Medium, Hard, Extreme

Unique Solution Assurance:

ensuring each generated puzzle has a unique solution

Validation Option:

enable players to compare their solutions with the correct one, highlighting errors

Hints and Complete Solutions:

hints and full solutions to maintain game accessibility across different skill levels

Software Engineering

Python & React Native

Expo Framework

UML design & User Stories

Thank you for your attention!

References:

[1] Lee, W.M., 2006. *Programming Sudoku*. Apress.

[2] Image Source:
https://codeahoy.com/learn/recursion/ch10/

[3] Fowler, M. and Highsmith, J., 2001.
The agile manifesto. Software development, 9(8), pp.28-35.