Sudoku Genius

System requirements :

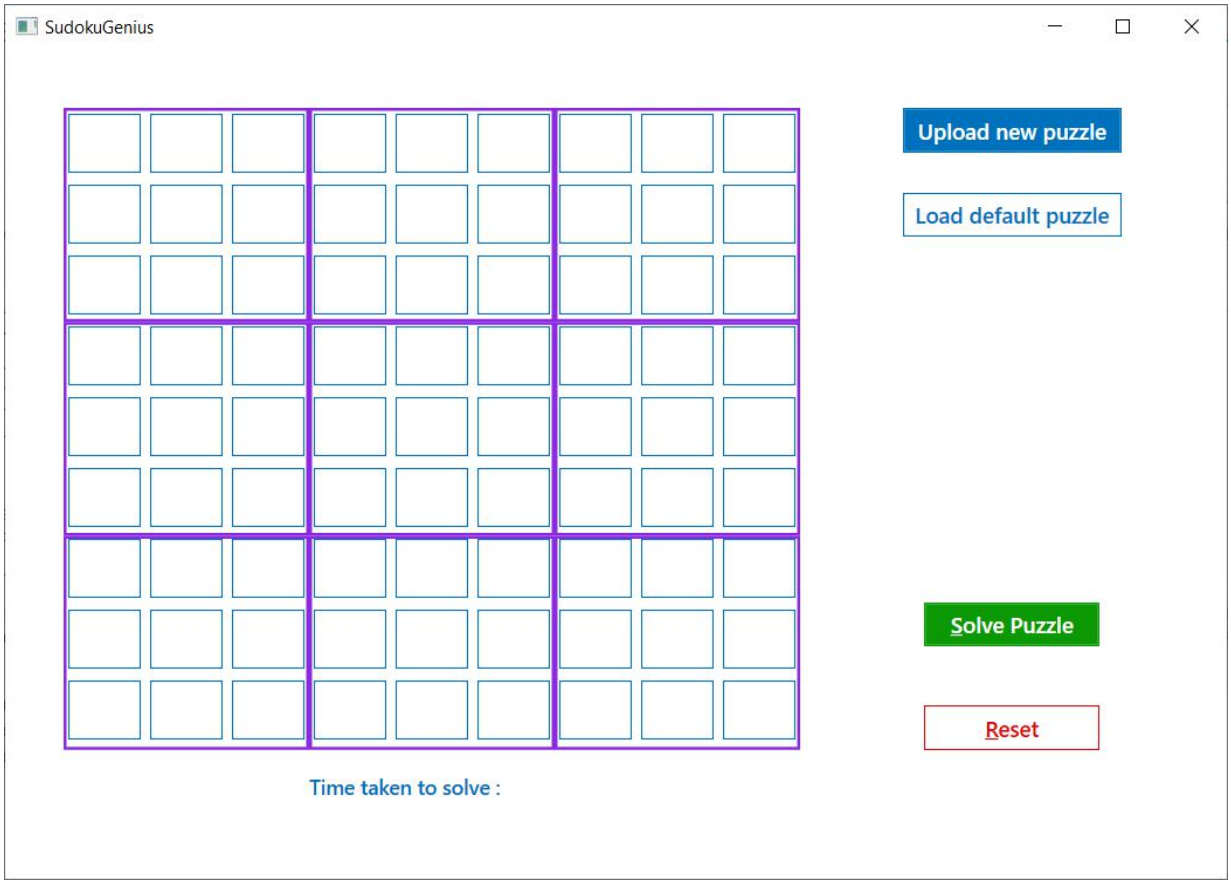
OS : Windows 7 or above. (.Net 4.5 supported OS)

External Dependencies(dlls) :

None. (no external package required)

How to Launch ?

Inside the SudokuGenius/Ouput folder, SudokuGenius.exe can be found. Please double click the exe then the application will be launched as shown below,



Levels

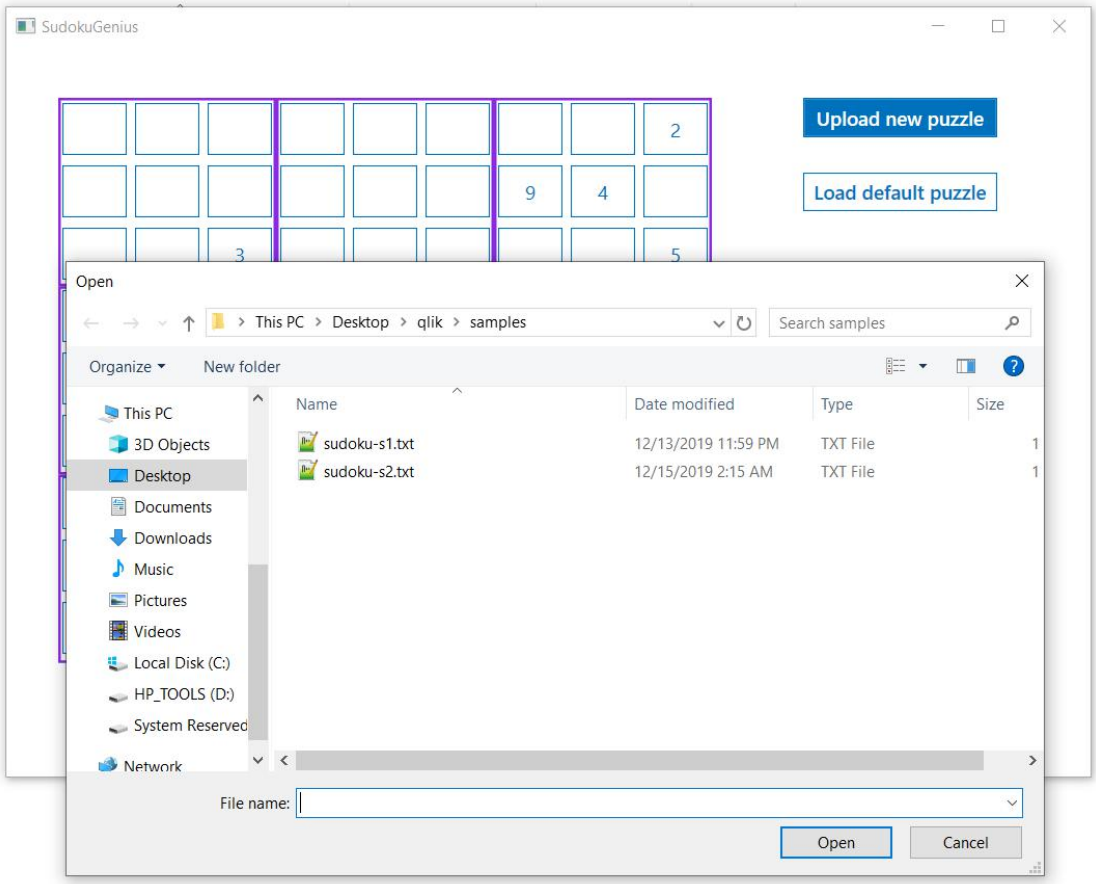
As this application is capable of handling any hard puzzle, didn't add any game levels to it.

Ways to play this puzzle

1. Upload new puzzle.
2. Load default puzzle.
3. Solve the puzzle with empty cells.

1. Upload new puzzle

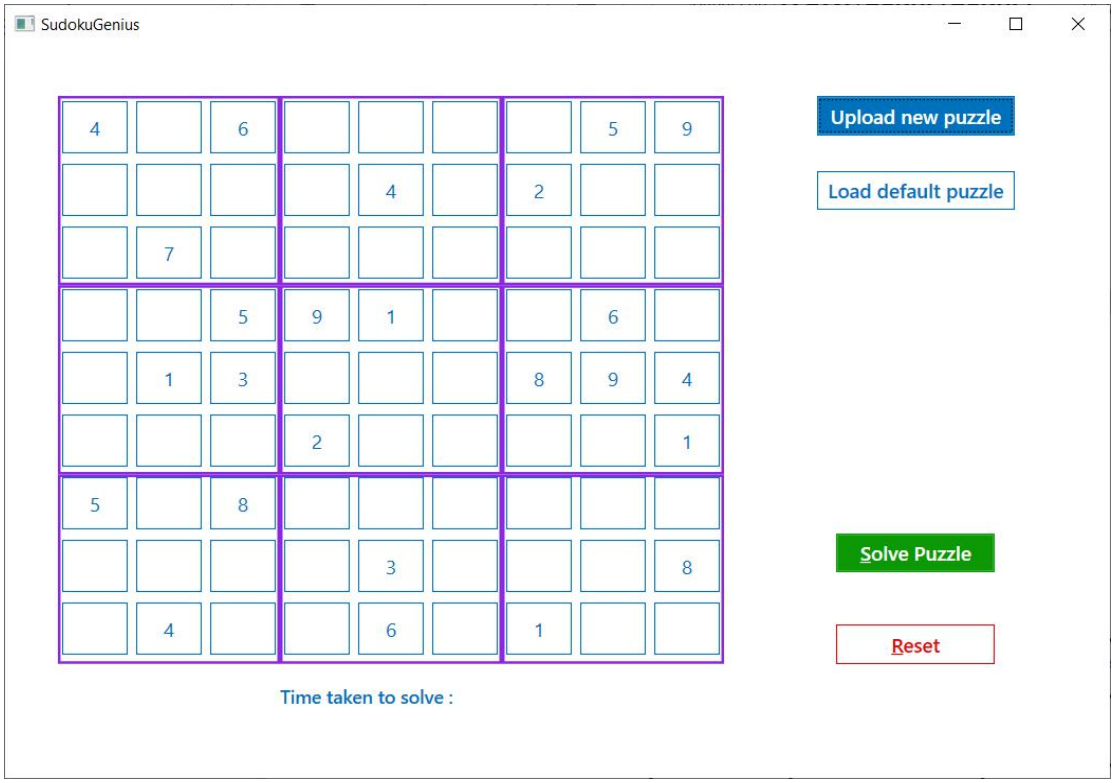
By clicking on “Upload new puzzle” button it open file picker. Then select the .txt input file which contains the puzzle.



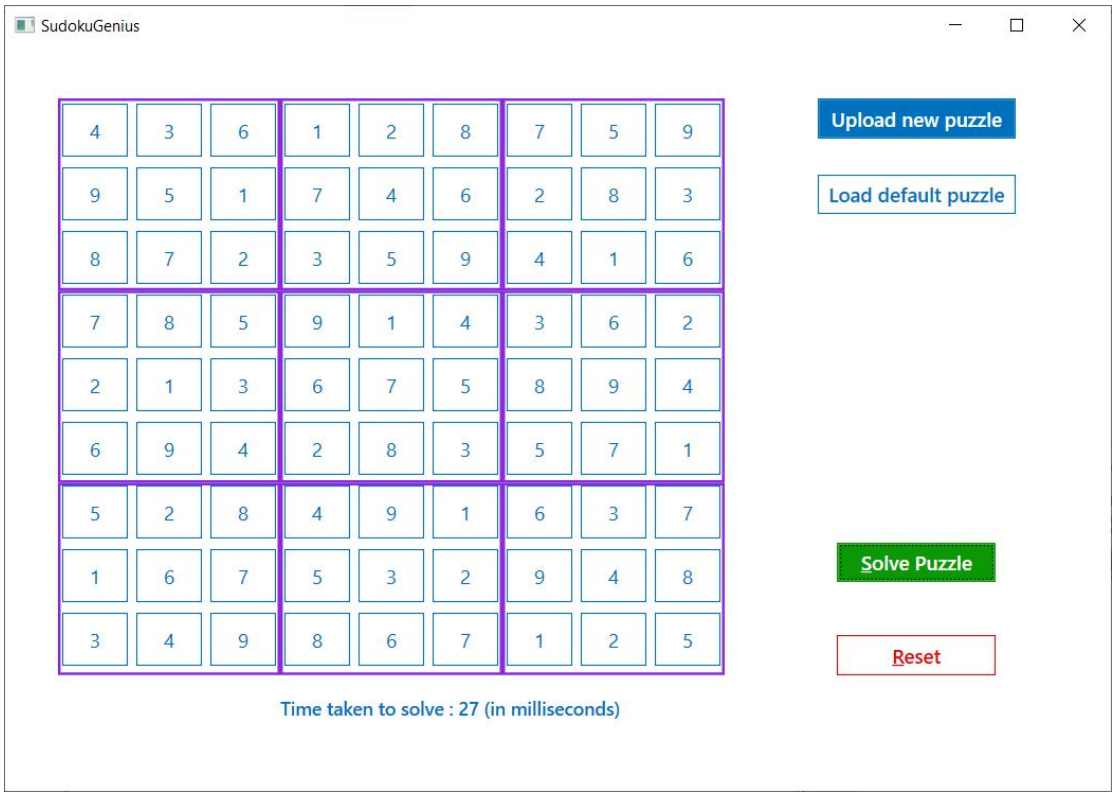
Puzzle should be in the below format,

4,0,6,0,0,0,0,5,9
0,0,0,0,4,0,2,0,0
0,7,0,0,0,0,0,0,0
0,0,5,9,1,0,0,6,0
0,1,3,0,0,0,8,9,4
0,0,0,2,0,0,0,0,1
5,0,8,0,0,0,0,0,0
0,0,0,0,3,0,0,0,8
0,4,0,0,6,0,1,0,0

Here 0 means empty cell. **Each cell value should be separated by comma**. Once the file is successfully added, puzzle will be imported to application.

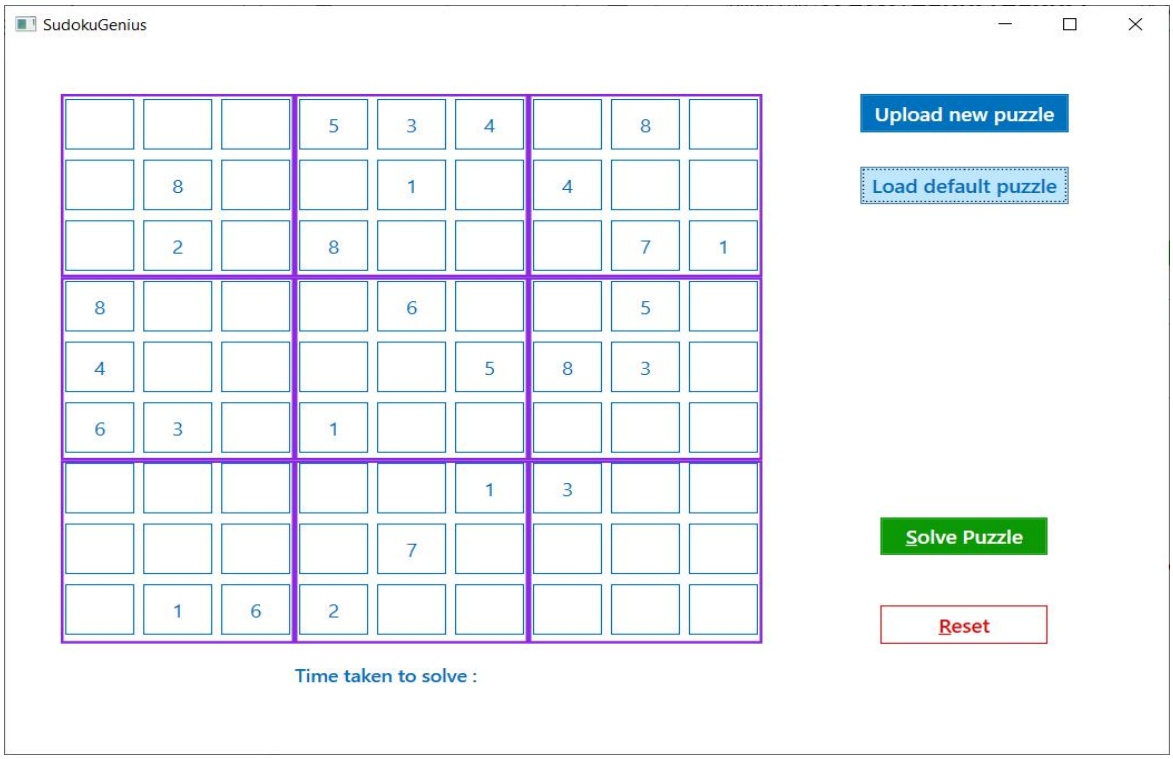


Then click on the solve button. It solves the puzzle in milli seconds.

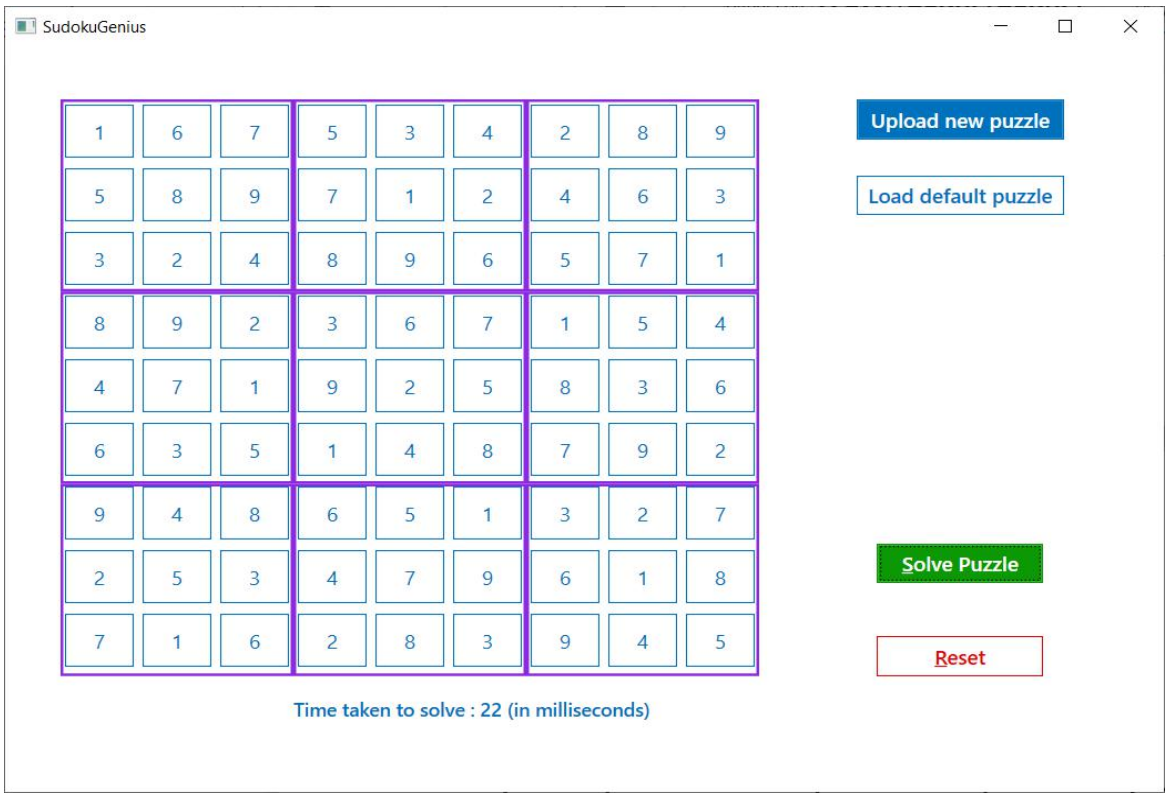


2. Load default puzzle

By clicking on the “Load default puzzle” button, it will add one default puzzle which is created by the application itself.

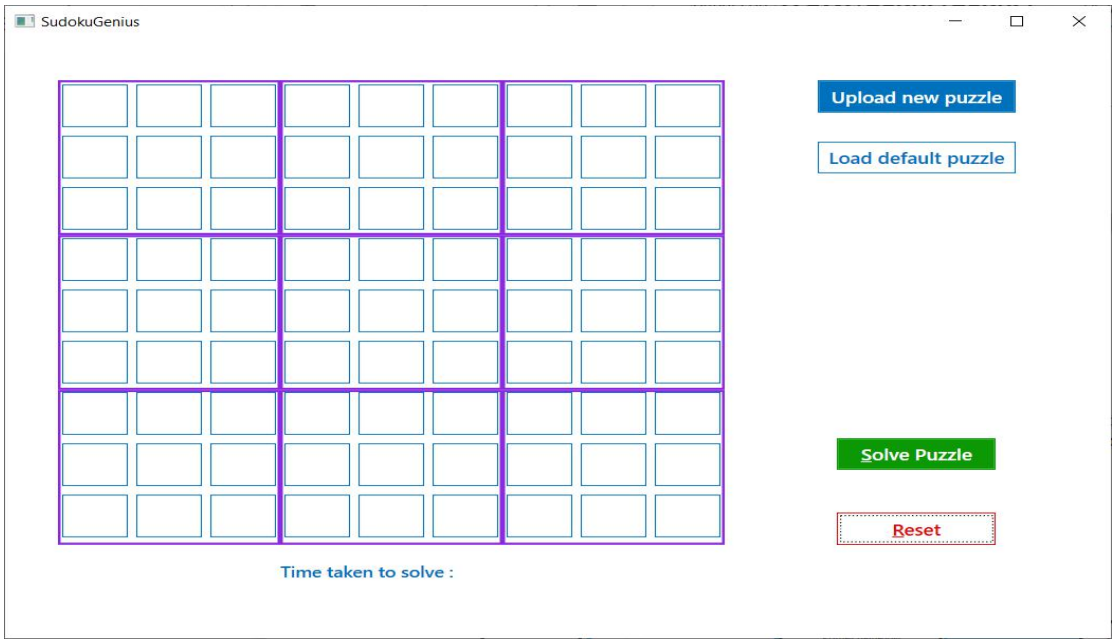


Then click on the solve button. It solves the puzzle in milli seconds.

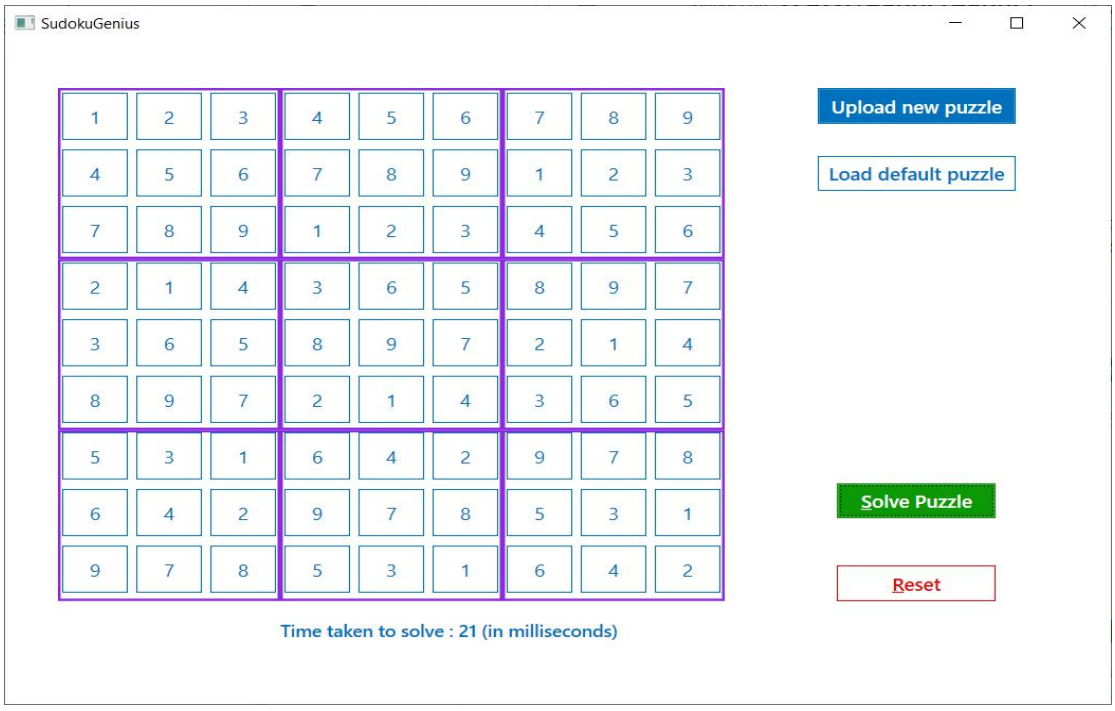


3. Solve the puzzle with empty cells

This application is capable of resolving puzzle with empty cells. Yes, without loading any puzzle if the solve puzzle button is clicked, it'll find the sudoku board for all the empty cells as shown below,



Below after solve button is clicked,

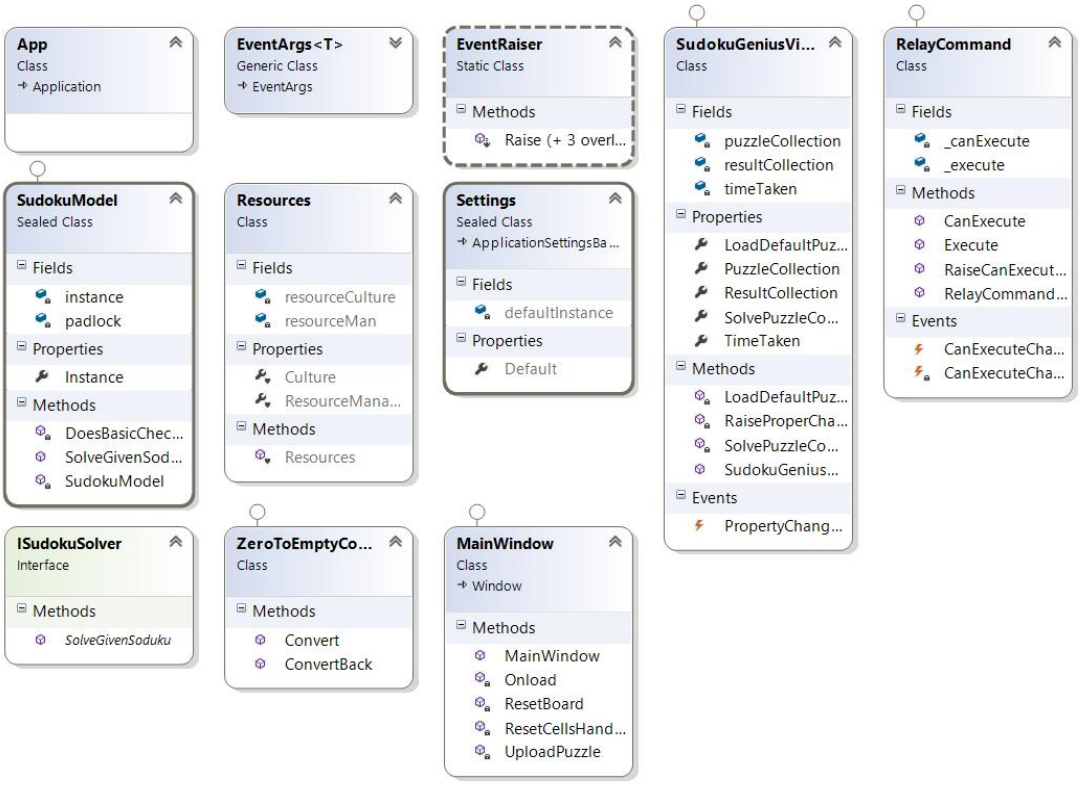


At any time, if Reset button is clicked, then all the cells will become empty.

Solution approach

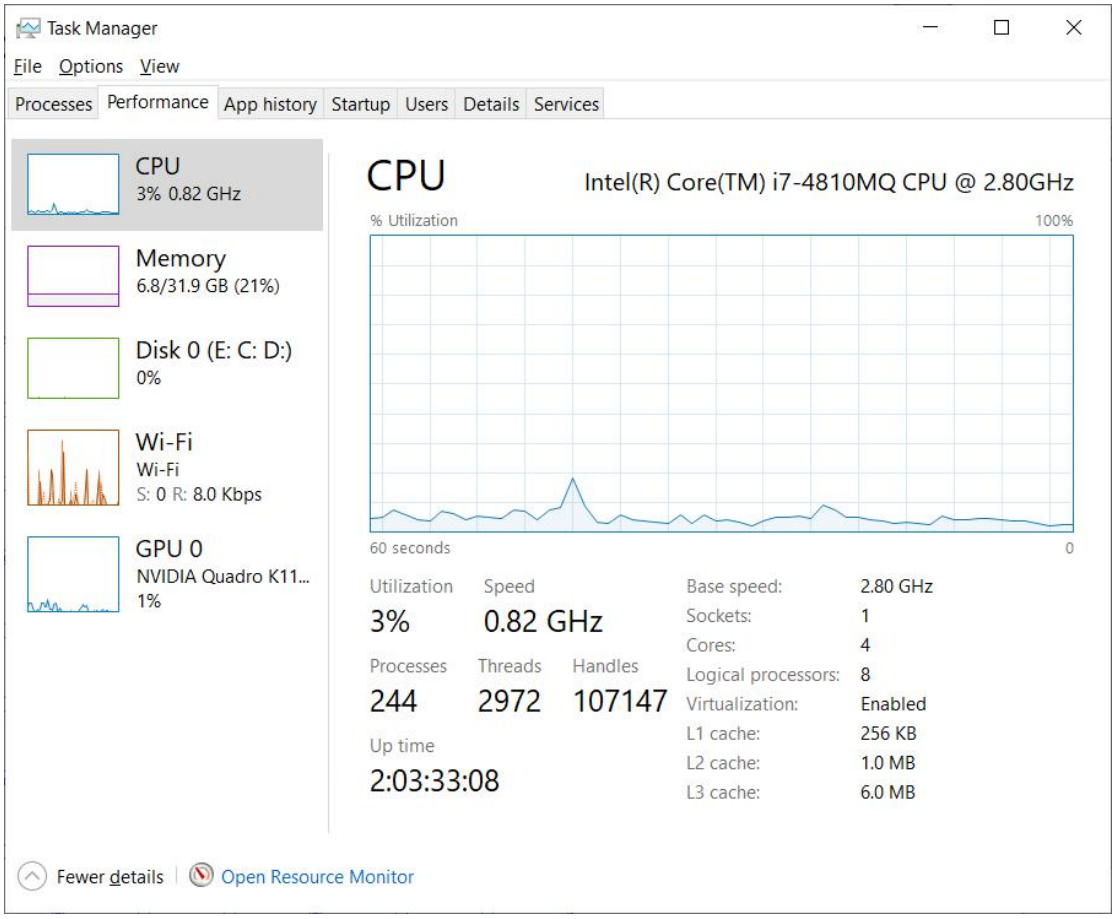
Architecture followed	:	MVVM (Model-View-ViewModel) for lightly coupled code
Technology	:	.Net 4.5, C#, WPF
IDE	:	Visual studio
Principle	:	SOLID
Design patterns used	:	Singleton pattern, Command pattern for GUI (using ICommand interface) Adapter pattern for ViewModel to Model communication.
Algorithm used	:	Backtrack algorithm (https://en.wikipedia.org/wiki/Backtracking)

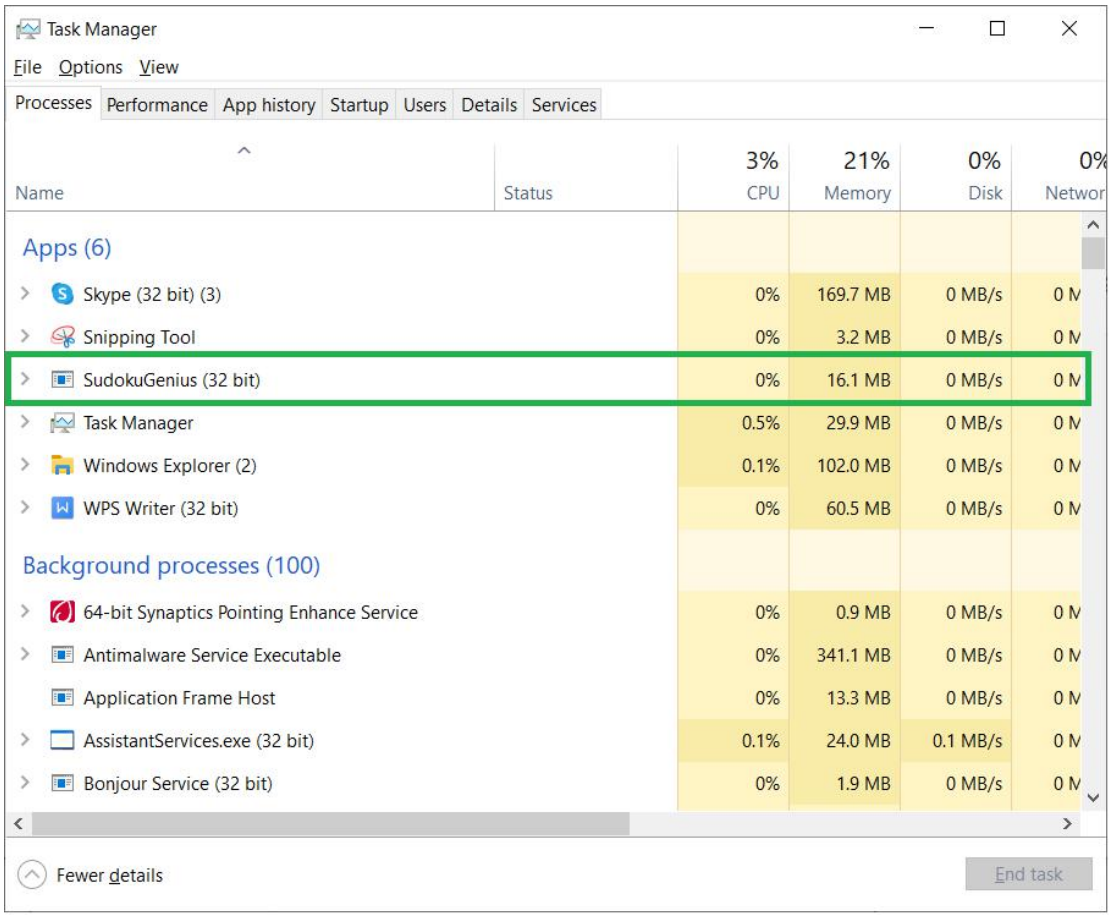
Class diagram



Code complexity and performance

Code complexity - O(N^2)
Average time taken - ~25 ms , cpu usage 0.5%


Task Manager
File Options View
Processes Performance App history Startup Users Details Services
CPU
3% 0.82 GHz
Memory
6.8/31.9 GB (21%)
Disk 0 (E: C: D:)
0%
Wi-Fi
Wi-Fi
S: 0 R: 8.0 Kbps
GPU 0
NVIDIA Quadro K11...
1%
CPU
Intel(R) Core(TM) i7-4810MQ CPU @ 2.80GHz
% Utilization
100%
60 seconds
0
Utilization
3%
Speed
0.82 GHz
Processes
244
Threads
2972
Handles
107147
Up time
2:03:33:08
Base speed: 2.80 GHz
Sockets: 1
Cores: 4
Logical processors: 8
Virtualization: Enabled
L1 cache: 256 KB
L2 cache: 1.0 MB
L3 cache: 6.0 MB
Fewer details
Open Resource Monitor


Task Manager
File Options View
Processes Performance App history Startup Users Details Services
Name
Status
3% CPU
21% Memory
0% Disk
0% Networ
Apps (6)
Skype (32 bit) (3) 0% 169.7 MB 0 MB/s 0 M
Snipping Tool 0% 3.2 MB 0 MB/s 0 M
SudokuGenius (32 bit) 0% 16.1 MB 0 MB/s 0 M
Task Manager 0.5% 29.9 MB 0 MB/s 0 M
Windows Explorer (2) 0.1% 102.0 MB 0 MB/s 0 M
WPS Writer (32 bit) 0% 60.5 MB 0 MB/s 0 M
Background processes (100)
64-bit Synaptics Pointing Enhance Service 0% 0.9 MB 0 MB/s 0 M
Antimalware Service Executable 0% 341.1 MB 0 MB/s 0 M
Application Frame Host 0% 13.3 MB 0 MB/s 0 M
AssistantServices.exe (32 bit) 0.1% 24.0 MB 0.1 MB/s 0 M
Bonjour Service (32 bit) 0% 1.9 MB 0 MB/s 0 M
Fewer details
End task

Improvements

1. Uploading file with different puzzle input formats and file validation.
2. Game levels
3. Index based puzzle solving.
4. Look for any performance loop hole.
5. More condition statements.