OS 2 -Programming Assignment -1(Validating Sudoku Solution)

Jakkala Naga Rohith Kumar(CS23BTECH11021)

January 2025

1 Introduction

As part of this assignment, need to implement validating sudoku by using the threads.

- The program should run in three different types sequential, chunk and mixed.

2 Libraries used

- pthread.h - for creating threads
- chrono - for calculating time
- fstream - for file handling
- matplotlib - for graphs creation
- subprocess - for automation of testing

3 File Structure

- Assgn1Src-cs23btech11021.cpp - main source file
- Assgn1Readme-cs23btech11021.txt - readme file contains explanation on how to use scripts in the folder.
- inp.txt - input file
- generategraph folder- contains python file for generating a graph from data
- makefile - to automate compilation and execution
- test.py - to run the executable for the test inputs
- tests folder- contains test inputs, outputs and input generators and times for the test input files

4 Data collection for graphs

- the input files is divided into different folders
- the input data is created in tests/generateinputs folder using cpp files
- the input files goes to the appropriate folders
- then the test.py will automate the program execution with these input files and take the output files and times in the test folder
- the executable is executed by same input 25 times and stores 25 outputs and timings stored in tests folder.
- these process done for all input files by test.py script
- finally in creation of graph we take these 25 timings and take average and create graph
- the graph can be created using graph.py file in generategraph folder.
- see the tests folder for inputs and outputs and times

5 Graphs

5.1 Experiment - 1

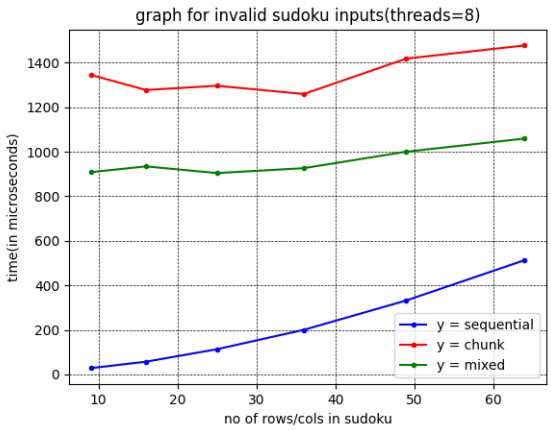

Figure 1: graph b/w time vs no of rows/cols in sudoku for invalid inputs

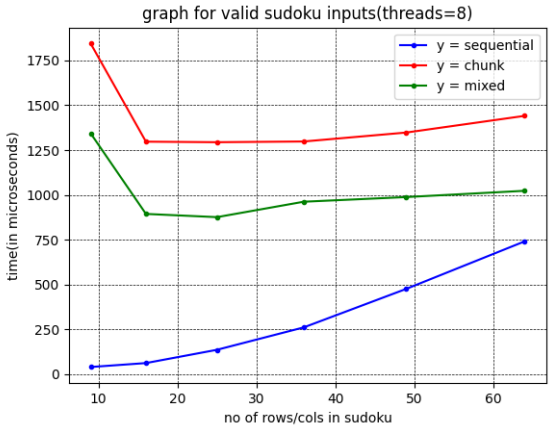

Figure 2: graph b/w time vs no of rows/cols in sudoku for valid inputs

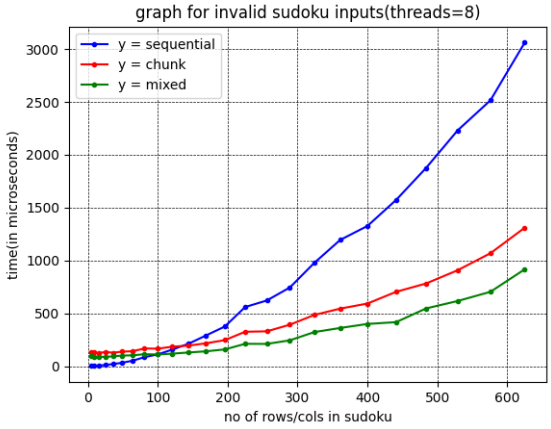

Figure 3: graph b/w time vs no of rows/cols in sudoku for invalid inputs with more inputs

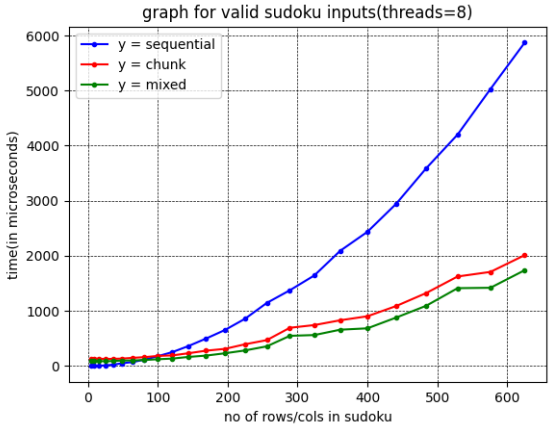

Figure 4: graph b/w time vs no of rows/cols in sudoku for valid inputs with more inputs

5.2 Experiment -2

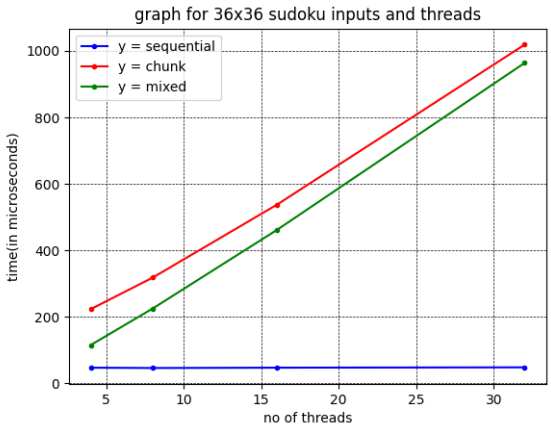

Figure 5: graph b/w time vs no of threads in sudoku for invalid inputs

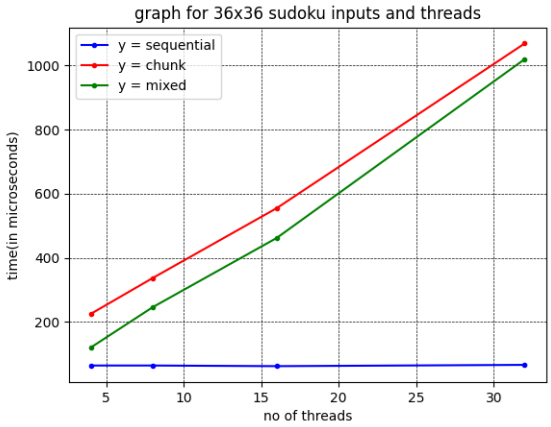

Figure 6: graph b/w time vs no of threads in sudoku for valid inputs

6 Observations

- we can observe a similar trend for (Figure 3 and Figure 4)(invalid and valid input in constant no of threads).
- at lower regions the overhead for the creating of the threads causes the chunk and the mixed to be slower than the sequential.(Figure 1 and Figure 2)
- but as the size of input increases the difference between them decreases.
- and after some point (here almost at 144x144) the sequential will become slower than chunk and mixed.
- chunk and mixed are almost at the same level, but the chunk seems to be slightly slower than the mixed.
- It may be because of the race conditions(rows and cols and grids can access an element at same time in the chunk) or cache effect (sequential access is more beneficial)
- we can observe the same trend in the Figure 3 and Figure 4
- As the no of threads increases for the same size it increases the time as the overhead for the thread creation increases.
- may be if the size of the Sudoku is large. It may go down and again increase after some point.

7 conclusion

- threading is useful if we perform independent tasks simultaneously.
- and threads may be useless if the input size is small.it may even increase the time due to creation overhead.
- using threads in mixed seems to be more beneficial than the chunk as it is less complex to write and easy to understand.
- Using sequential in low-range inputs is beneficial
- more threads in the program is also not beneficial
- performance of threads also depends on the machine in which program is executing