Developing Parallel Algorithms for Creating and Solving Sudoku Puzzle

Mahmoud Sayed and Ugur Koc
mfayoub@cs.umd.edu, ukoc@cs.umd.edu
Department of Computer Science, University of Maryland College Park

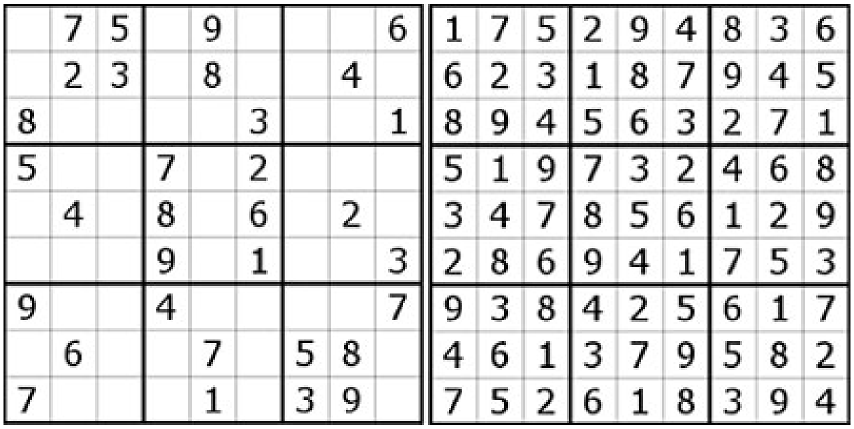
Abstract: In this project, we will implement parallel algorithms to create and solve Sudoku puzzle. Our goals are 1) to compare effectiveness and efficiency of algorithms and different heuristic functions and 2) to measure speedups of our algorithms against their sequential versions on these two highly related tasks (i.e. creating a Sudoku initial board and solving it).

Introduction

Sudoku is a puzzle on 9×9 two dimensional game board, i.e. 81 cells. When the game begins some of the cells are filled with numbers ranging from 1 to 9. The goal of the game is to fill each cell with a number from this range satisfying following three constrains: do not have any duplicate numbers in 1) a row, 2) a column, and 3) a box bordered in bold as in Figure 1. In addition to these constraints, for any initial board configuration, there should be only one unique solution (needed to be considered when creating a initial configuration).

Sudoku can be formed as a search problem for which we can apply different kind of algorithms, e.g. greedy search, local search, graph search. Many of these algorithms rely on a heuristic cost function which can be modeled as the number of possibilities left to solve the whole board in our case.

The goals of our project is to explore implementing search algorithms in parallel, and we took solving Sudoku puzzles as an application to measure speedups if any. Below, we list our objectives, but we are not sure of the time constraint.



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

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

Figure 1: A sample Sudoku puzzle and its solution (*1*)

- We will examine different algorithms and see the speedup achieved if run in parallel.

- There are a lot of strategies that can help in filling some blank cells at the beginning, which will result in minimizing the size of the problem. We need to study how good these strategies are and whether they can be parallelized or not. A simple strategy is to consider only one number and it targets to finish by filling that number in each row, column, and box without violating game rules. Obviously, we can do this in parallel each of which is working on a different number.

- We will investigate on a complementary problem which is how to create Sudoku puzzles in parallel. We think, this problem is not trivial because we need to provide different puzzle difficulties. A difficult board has less numbers filled in the initial configuration. Additionally, for any generated board, we need to guarantee that there is only one possible solution for it. Relating to some work (*3*), a possible way is to have a complete solved board, and then we can remove some elements of it.

Implementation

We will realize this project in c programming language. Based on the natures of the search algorithms and heuristic functions, we will use OpenMP, OpenMPI, and maybe pthread libraries to parallelize them.

One important concern of us is to have generic implementations which then can be used to solve other kind of puzzle games.

Project management

This project consists of three phases. In the first phase, we will research more about puzzle solvers and parallel programming to improve our current understanding and to be aware of the state-of-the-art (*2*–*5*). This may take around one and a half week. Second phase is the design and implementation phase. Using the understanding we gathered from phase 1, we will design and implement necessary algorithms in parallel and build tools to realize the project. This phase is the heart of the project. Thus, we plan to spend more time for it -roughly three weeks. Last phase is the experimentation. To evaluate the efficiency and effectiveness of our algorithms and tools, we will conduct set of experiments which we will design considering different scenarios. This phase may take roughly one and a half week too.

Since we are a team of two, collaboration is relatively easy for us. Throughout the time-line of the project, we will always be working together. There will be small tasks which will be assigned to individuals, but each of us we will be involved in all three phases.

References and Notes

1. A sample sudoku puzzle and its solution. `https://www.sciencenews.org/sites/default/files/9416`.

2. Alton Chiu, Ehsan Nasiri, and Rafat Rashid. Parallelization of sudoku. Technical report, University of Toronto, 2012.

3. William Dudziak. Arbitrary size parallel sudoku creation. Technical report, 2006.

4. Hilda Huang and Lindsay Zhong. Parallel sudoku solver. Technical report, Carnegie Mellon University, 2013.

5. Rohit H. Shreekant. Parallel sudoku solver using mpi and c. Technical report, State University of New York at Buffalo, 2014.