CS 21 Machine Problem
Sudoku Solver

Department of Computer Science
College of Engineering
University of the Philippines - Diliman

1 Introduction

For this Machine Problem, you will be creating a program in MIPS assembly that solves a Sudoku board. The input is a board with empty cells and the output is the solution for the board. The main requirement is a 4x4 Sudoku solver. You may also implement the traditional 9x9 Sudoku solver for additional points.

1.1 Sudoku

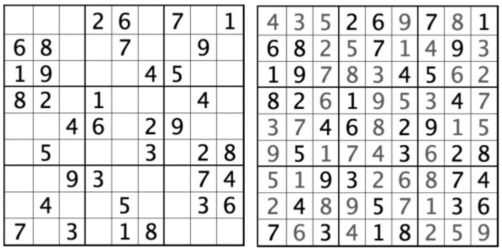
The original Sudoku involves a 9x9 grid which is further divided into 9 3x3 divisions. The numbers 1 to 9 must appear exactly once in each row, each column, or each 3x3 division. An example problem can be seen in Figure 1.

Simplifying Sudoku into a 4x4 grid, you have the numbers 1 to 4 which must appear exactly once in each row, each column, or each 2x2 division. An example problem can be seen in Figure 2.

1.2 Backtracking

In order to solve the Sudoku board, you may use backtracking. Backtracking is a search technique which involves checking each possible state of the problem in order to find the solution. Figure 3, shows how all the possible states in order to solve the 4-Queens Problem through backtracking (figure taken from here).

Here, the problem starts at state 0. It will then go to state 1. Since state 1 is a valid state, it will proceed further to state 2. State 2 violates the constraints of the problem, so it will go back to state 1. From state 1 again, it will check state 3. Similarly, state 3 is invalid so it goes back to state 1 and now proceeds to state 4. State 4 is a valid state so it will check state 5. This goes on until the solution has been found (marked with a check in the figure).



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

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

Figure 1: 9x9 Sudoku board with solution on the right

2	1		
	3	2	
			4
1			

2	1	4	3
4	3	2	1
3	2	1	4
1	4	3	2

Figure 2: 4x4 Sudoku board with solution on the right

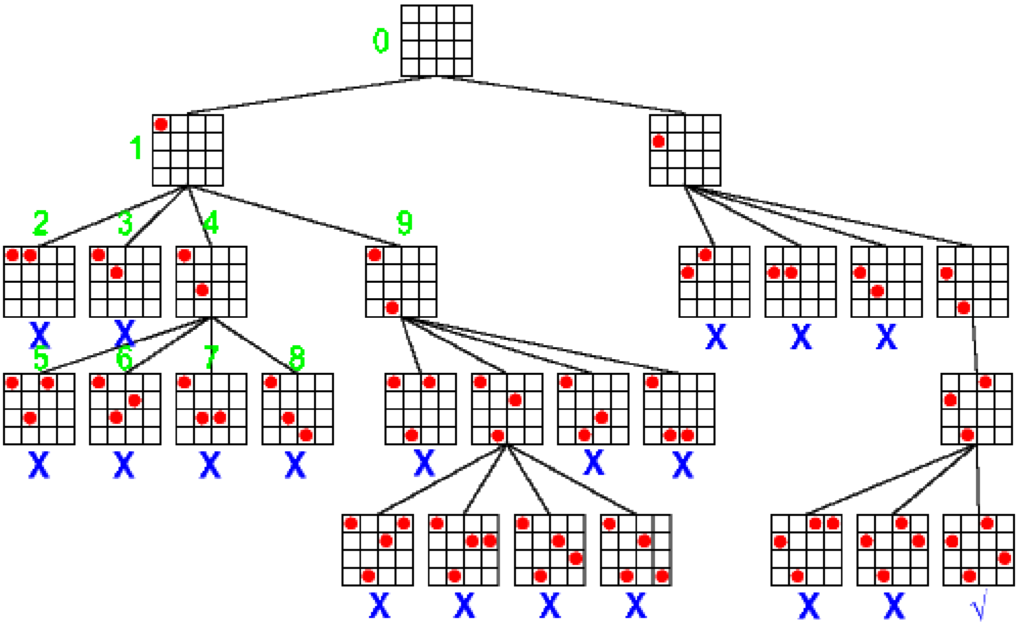


Figure 3: Backtracking example

2 Input and Output

For both Sudoku variants, input will consist of lines of integers corresponding to a row in the Sudoku board. Cells that are empty will be represented by 0. It is assumed that for the 4x4 solver, there will always be four input lines consisting of four integers for each line. For example, the problem input for Figure 2 will look like the following:

```
2100
0320
0004
1000
```

The solution for this board must look like the following:

```
2143
4321
3214
1432
```

For the 9x9 solver, it is assumed that there will always be nine input lines consisting of nine integers for each line. For example, the problem input for Figure 1 will look like the following:

```
000260701
680070090
190004500
820100040
004602900
050003028
009300074
040050036
703018000
```

The solution for this board must look like the following:

```
435269781
682571493
197834562
826195347
374682915
951743628
519326874
248957136
763418259
```

3 Point System

The maximum score for the Machine Problem is 120/100 and is composed of the following:

- Working 4x4 solver - **70 points**
- Working 9x9 solver - **50 points**

Note that the 4x4 solver must be working before you can get points for the 9x9 solver. The documentation must also exist and be submitted on time as a requirement for checking the solver(s).

4 Documentation

Submit the documentation in PDF form with your student number as the filename (e.g. `202012345.pdf`). The documentation must contain the following:

1. A summary of how the solver(s) work.
2. A high-level pseudocode of the algorithm you implemented for the solver(s).
3. An in-depth explanation of each function in the 4x4 solver and how they are used to solve the problem.
4. An in-depth explanation of each function in the 9x9 solver and how they are used to solve the problem (if applicable).
5. Sample test case(s) for each solver aside from the examples given in this document.

Aside from the PDF documentation, you must also submit a video demo with your student number as the filename (e.g. `202012345.mp4`), explaining how the solver works. Try out test cases as well. This is basically a video version of the PDF documentation. You may directly submit the video file into the UVLE submission bin if it is under the size limit. If not, you may upload it to Google Drive and include the link either in the PDF documentation or in the text field included in the submission bin. Note that the timestamp of the video demo must also be before the deadline.

Nonsubmission or late submission of PDF and video documentation automatically results in a grade of zero.

5 Submission

Submit the 4x4 solver with the filename as your student number appended by underscore 4 (e.g. `202012345_4.asm`). Submit the 9x9 solver with the filename as your student number appended by underscore 9 (e.g. `202012345_9.asm`). Together with the assembly files, include the PDF and video documentation in the appropriate UVLE submission bin. The deadline for the machine problem will be listed in UVLE.