Sudoku : A constraint satisfaction problem

Aditya Joshi, Barada p Acharya, Bhushan Pagariya, Vinay Bagade
University of California, Irvine

Abstract

In this report, we describe Sudoku as a constraint satisfaction problem. CSP is viewed as consisting of four major components namely the variable heuristic, value heuristic, domain reduction and early detection, this report explores the efficiency of various methods used in each of the above components and displays statistical results for the same.

A. Introduction

Sudoku is a puzzle where the player is given an 9X9 grid and some of the cells in the grid are assigned a value the player has to fill up the grid in such a way that in a row, column and a box of 3X3 numbers appear with no repetition. Each cell can take values in {1,2,...9}. The literal meaning of Sudoku in Japanese is 'single number' and the puzzle is among the most commonly featured puzzles in newspapers and magazines.

B. Problem Description

Variables: Xij is a variable corresponding to the cell Mij of the matrix, for 1 <= i,j <= 9

Domains: Domains of each Xij corresponds to {1,2,3….9}

Constraints: alldiff constraints between elements of the same row, column and a square matrix of 3X3.
Note: The square matrix of 3X3 is not any 3X3 subset but one which has its first cell on a row k where k % 3 = 1.

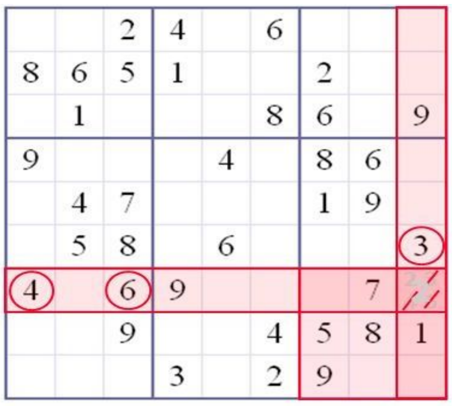

Figure1: Example problem and solution

C. Input

The input is a sequence of 81 integers in the range 0 to 9 given in row major format. 0 corresponds to the empty cell.

Data Structure: The input is stored in an adjacency list with every row corresponding to a cell. The adjacency of list will always contain the allowed values corresponding to that cell. If the solution exists and is found every variable consists of one value is the list.

D. Our Solver

The three important steps to solve a constraint satisfaction problem(CSP) are:

1. Selection of variables
2. Assigning value to the variables
3. Constraint Propagation and backtracking

1. Selection of variable

While solving any CSP, one of the critical thing is the order in which variables are selected. We are using Most Restricted Variable (MRV) and Maximum Degree (MD) variable selection heuristics in our project.

Most Restricted Variable (MRV)

MRV heuristic selects variable having least possible legal values in its domain. This heuristic is also known as 'fail-first' heuristic because it picks variable that is most likely to cause failure soon, thereby pruning the search tree. If some variable X has no legal value left, the MRV will select X and failure will be detected immediately.

Maximum Degree (MD)

If there are two variables having least possible legal values in their domain. So in this scenario we are Maximum Degree heuristic as a tiebreaker. MD heuristic will select a variable having largest number of constraints on other unassigned variables. MD heuristic attempts to reduce branching factor on future choices by selecting variable having maximum branching factor in early iteration.

2. Assigning value to the variables

Once the variable is selected, next important thing is to assign a value to that variable. We are using Least Constraining Value (LCS) heuristic and Probabilistic heuristic(PR) to decide order in which values are selected.

Least Constraining Value (LCS)

LCS heuristic selects the value which rules out the fewest choices from the neighbouring unassigned variable. In general, LCS heuristic is trying to leave the maximum flexibility for subsequent variable assignment.

Probabilistic Heuristic (PR)

Probabilistic heuristic selects a value which is most likely to be a valid assignment. We are using following formula for computing probability of each value and then we are choosing a value having highest probability.
Probability of a value K to be a valid assignment = ((9 - X)/9
where X : the number of occurrences of value K in assigned variables

3. Constraint propagation and Backtracking

Naive Backtracking (BT)

A number (1 to 9) is randomly selected and before assigning it to a variable, we do a constraint satisfaction check, then we check recursively that this assignment will lead to a solution or not. If the assignment won't lead to a solution, then we try next number for the variable. And if none of the number (1 to 9) leads to the solution, we return false.The problem with this approach is that it takes a lot of time. We can improve on this naive approach by doing some sort of preprocessing and constraint propagation. Some of the approaches are discussed below.

Forward Checking (FC)

The Forward checking algorithm is the most elementary form of the constraint propagation. It eliminates in advance, the possibilities that do not match the constraints from the domains of unassigned variables connected to the variable whose values is being assigned. For example, if we assign the digit '1' to cell c1, we eliminate '1' from all variables in the corresponding row, column and box.

Arc Consistency (AC)

The arc consistency is a kind of preprocessing step before the actual backtracking. What we mean by this is that if two variables connected by an arc in constraint graph are arc consistent then each of admissible values of one variable is consistent with some admissible value of the other variable.
Generally, arc consistency is implemented using a queue where all the arcs of a constraint graph are first pushed into the queue and then the algorithm pops an arc from the queue and prunes the domain of the corresponding connected variables till they are consistent with each other and arcs corresponding to all the variables connected to the pruned variable are inserted back into the queue. This process is repeated while the queue is not empty.
What we implemented here in our project is not exactly arc consistency, but a flavour of the same. Here, in the project we recursively prune the domains of the unassigned variables which are connected to a variable whose domain has been reduced to one. The function is recursively called on any variables whose domain was reduced to one because of our previous assignment.

Early Detection (ED)

Another algorithm we have implemented for domain reduction during constraint propagation is Early detection. Here, after assigning the value to the variable we check for the corresponding row, column and box if the number of unassigned variables is more than the number of unique values they can take if that is the case then we stop immediately.

E. Observation

A sample collection of sudoku based upon difficulty level (like easy, medium and hard) is collected from the internet. Several combinations of the above-defined algorithm were applied upon problem set and the results have been analyzed.

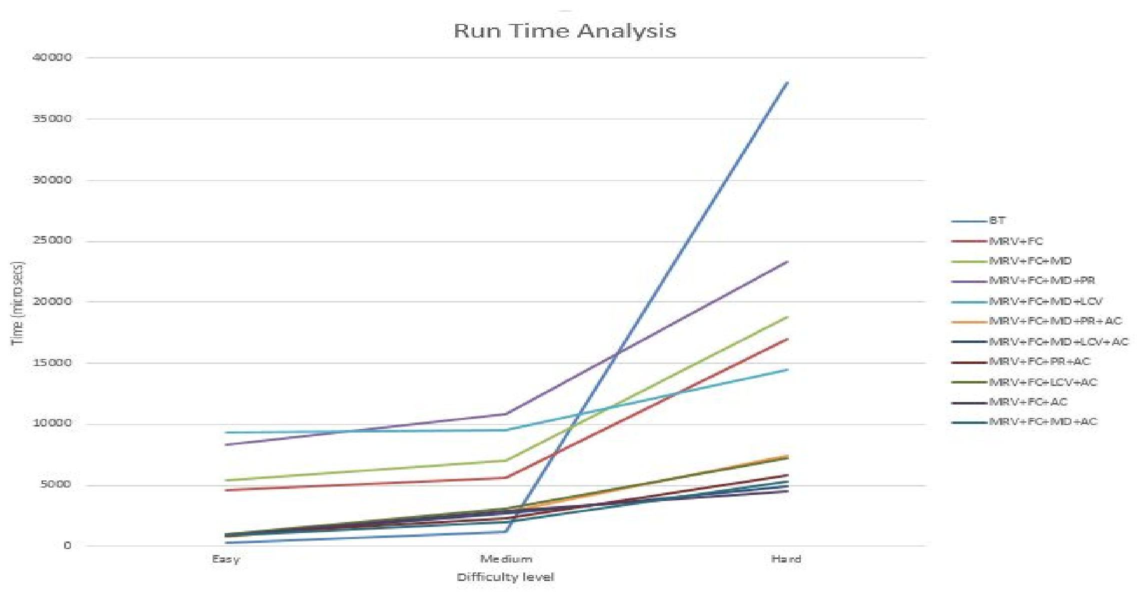

Figure 2: Run Time Analysis plot between algorithms BT, MRV+FC, MRV+FC+MD, MRV+FC+MD+PR, MRV+FC+MD+LCV, MRV+FC+MD+PR+AC, MRV+FC+MD+LCV+AC, MRV+FC+PR+AC, MRV+FC+LCV+AC, MRV+FC+AC, MRV+FC+MD+AC.

From the above graph, we can assert that run-time of an algorithm like brute-force backtracking search is exponentially increasing with difficulty level of the problem, However, CSP algorithms are maintaining a steady increase in run-time values. Among various combinations of CSP, we can see that MRV+FC+AC, MRV+FC+PR+AC, MRV+FC+LCV+AC algorithms are generating better results than other. So we will select these algorithms for our further study in the project.

F. Experiment Results

The experiments for all problems were run on a laptop with 2.4GHz, Core i5 processor and 4 GB of memory under Window 7.

Puzzle	Brute-force Backtracking		MRV+FC		MRV+FC+AC		MRV+FC+Prob+AC		MRV+FC+LCV+AC	
	Runtime (µs)	No. of backtracks	Runtime (µs)	No. of backtracks	Runtime (µs)	No. of backtracks	Runtime (µs)	No. of backtracks	Runtime (µs)	No. of backtracks
Easy	780	278	2322	20	775	4	1076	7	593	4
Medium1	3973	1924	4322	66	2346	14	2428	15	2595	19
Medium2	1593	789	2351	12	957	2	937	2	944	2
Hard1	1822	864	2664	0	1907	8	1593	8	1350	6
Hard2	3915	1626	1986	5	1509	4	1381	5	1301	4
Evil1	17479	9737	14105	340	3392	19	2375	14	12547	137
Evil2	83359	76285	11988	287	1776	5	5037	54	1316	4
Evil3	1237	1311	1986	0	4223	27	1778	23	1519	6

Table 1: Comparison of Brute-force backtracking, MRV+FC, MRV+FC+AC, MRV+FC+Prob+AC and MRV+FC+LCV+AC algorithms.

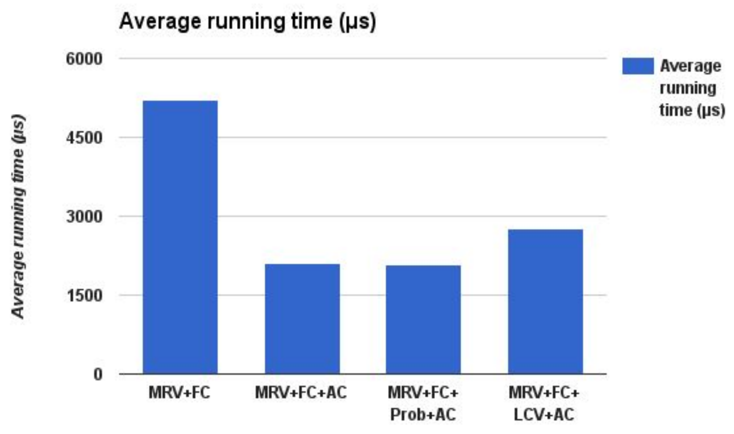

Figure 3(a): Runtime analysis of MRV+FC, MRV+FC+AC, MRV+FC+Prob+AC and MRV+FC+LCV+AC algorithms

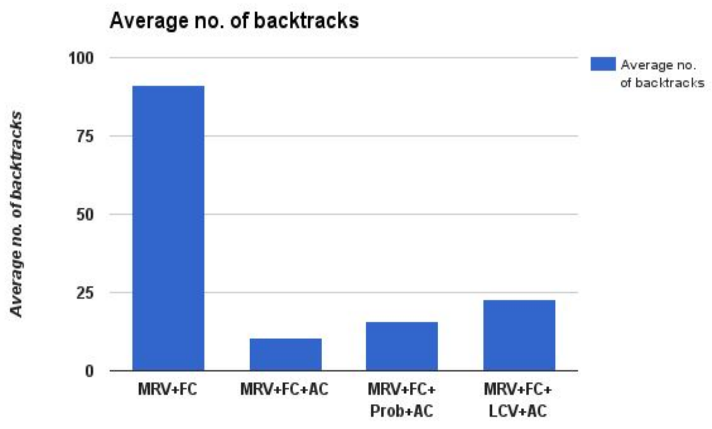

Figure 3b: Backtrack comparison of MRV+FC, MRV+FC+AC, MRV+FC+Prob+AC and MRV+FC+LCV+AC algorithms

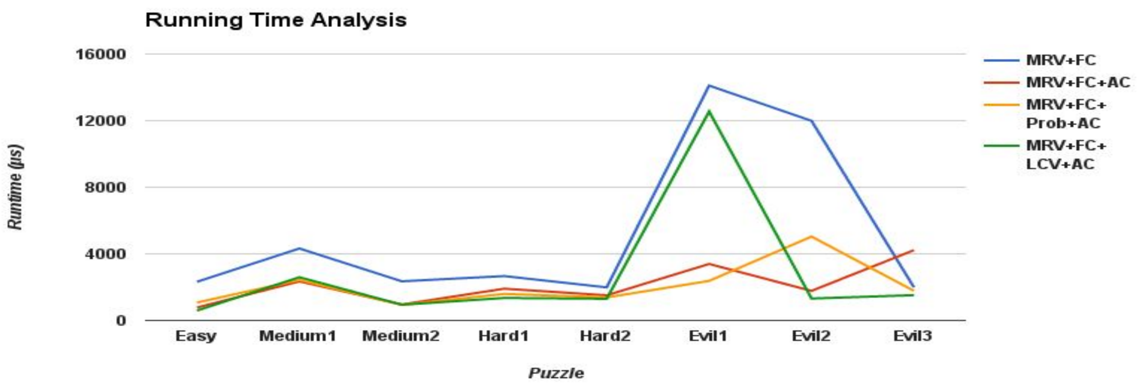

Figure 4: Runtime analysis of MRV+FC, MRV+FC+AC, MRV+FC+Prob+AC and MRV+FC+LCV+AC algorithms for different problems

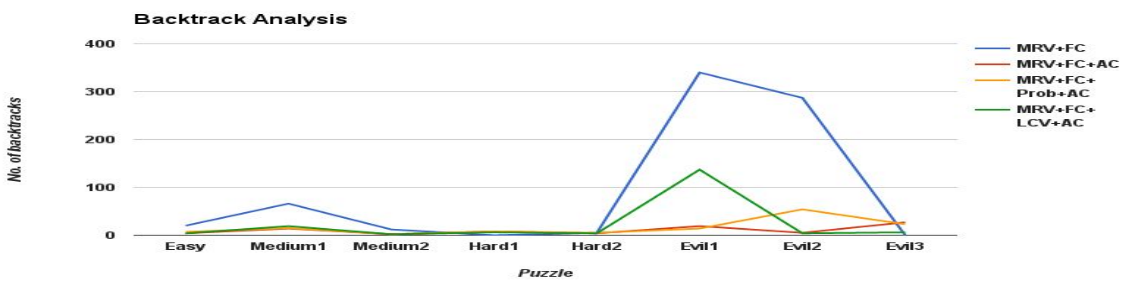

Figure 5: Backtrack analysis of MRV+FC, MRV+FC+AC, MRV+FC+Prob+AC and MRV+FC+LCV+AC algorithms for different problems.

G. Conclusion

Based on the run-time analysis, we can infer that algorithm using CSP takes less run-time than brute-force Backtracking approach. These gains are seen due to preprocessing of data for domain reduction and early detection failure scenario. We can see that CSP using arc-consistency generally gives better solution CSP without arc-consistency. Our algorithm provides an efficient way to solve the sudoku puzzle by formulating the puzzle as a Constraint Satisfaction Problem and applying concepts such as arc-consistency, forward checking , constraint propagation, early detection with using various value and variable heuristic.

References

[1] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach (3rd Edition)*

[2] Wikipedia page on Sudoku. https://en.wikipedia.org/wiki/Sudoku

[3] Brute-force backtracking algorithm. http://www.geeksforgeeks.org/backtracking-set-7-suduku

Results for benchmark test-cases

Puzzle 1

Easy
0 6 4 0 0 3 8 5 1 2 0 0 0 0 0 9 6 0 0 0 5 8 0 0 0 0 0 0 2 0 0 4 1 0 9 0 0 3 6 0 5 0 1 4 0 0 1 0 2 6 0 0 3 0 0 0 0 0 0 4 2 0 0 0 5 8 0 0 0 0 0 9 6 4 2 1 0 0 5 8 0

MRV+FC	MRV+FC+AC:	MRV+FC+Prob+AC	MRV+FC+LCV+AC
9 6 4 7 2 3 8 5 1 2 8 3 4 1 5 9 6 7 1 7 5 8 9 6 3 2 4 5 2 7 3 4 1 6 9 8 8 3 6 9 5 7 1 4 2 4 1 9 2 6 8 7 3 5 3 9 1 5 8 4 2 7 6 7 5 8 6 3 2 4 1 9 6 4 2 1 7 9 5 8 3	9 6 4 7 2 3 8 5 1 2 8 3 4 1 5 9 6 7 1 7 5 8 9 6 3 2 4 5 2 7 3 4 1 6 9 8 8 3 6 9 5 7 1 4 2 4 1 9 2 6 8 7 3 5 3 9 1 5 8 4 2 7 6 7 5 8 6 3 2 4 1 9 6 4 2 1 7 9 5 8 3	9 6 4 7 2 3 8 5 1 2 8 3 4 1 5 9 6 7 1 7 5 8 9 6 3 2 4 5 2 7 3 4 1 6 9 8 8 3 6 9 5 7 1 4 2 4 1 9 2 6 8 7 3 5 3 9 1 5 8 4 2 7 6 7 5 8 6 3 2 4 1 9 6 4 2 1 7 9 5 8 3	9 6 4 7 2 3 8 5 1 2 8 3 4 1 5 9 6 7 1 7 5 8 9 6 3 2 4 5 2 7 3 4 1 6 9 8 8 3 6 9 5 7 1 4 2 4 1 9 2 6 8 7 3 5 3 9 1 5 8 4 2 7 6 7 5 8 6 3 2 4 1 9 6 4 2 1 7 9 5 8 3
**Time :- 2322 **Backtracks :- 20	**Time :- 775 **Backtrack:- 4	**Time :- 1076 **Backtracks :- 7	**Time :- 593 **Backtracks :- 4

Puzzle 2

Medium1
7 2 0 5 0 0 0 0 6 0 9 5 1 6 0 0 0 0 0 0 0 7 0 0 9 0 4 2 1 0 9 0 0 0 0 3 0 0 9 0 0 0 4 0 0 6 0 0 0 0 4 0 9 1 3 0 2 0 0 6 0 0 0 0 0 0 0 5 3 6 2 0 9 0 0 0 0 1 0 4 5

MRV+FC	MRV+FC+AC:	MRV+FC+Prob+AC	MRV+FC+LCV+AC
7 2 3 5 4 9 8 1 6 4 9 5 1 6 8 2 3 7 8 6 1 7 3 2 9 5 4 2 1 4 9 8 5 7 6 3 5 3 9 6 1 7 4 8 2 6 7 8 3 2 4 5 9 1 3 5 2 4 9 6 1 7 8 1 4 7 8 5 3 6 2 9 9 8 6 2 7 1 3 4 5	7 2 3 5 4 9 8 1 6 4 9 5 1 6 8 2 3 7 8 6 1 7 3 2 9 5 4 2 1 4 9 8 5 7 6 3 5 3 9 6 1 7 4 8 2 6 7 8 3 2 4 5 9 1 3 5 2 4 9 6 1 7 8 1 4 7 8 5 3 6 2 9 9 8 6 2 7 1 3 4 5	7 2 3 5 4 9 8 1 6 4 9 5 1 6 8 2 3 7 8 6 1 7 3 2 9 5 4 2 1 4 9 8 5 7 6 3 5 3 9 6 1 7 4 8 2 6 7 8 3 2 4 5 9 1 3 5 2 4 9 6 1 7 8 1 4 7 8 5 3 6 2 9 9 8 6 2 7 1 3 4 5	7 2 3 5 4 9 8 1 6 4 9 5 1 6 8 2 3 7 8 6 1 7 3 2 9 5 4 2 1 4 9 8 5 7 6 3 5 3 9 6 1 7 4 8 2 6 7 8 3 2 4 5 9 1 3 5 2 4 9 6 1 7 8 1 4 7 8 5 3 6 2 9 9 8 6 2 7 1 3 4 5
**Time :- 4322 **Backtracks :- 66	**Time :- 2346 **Backtrack:- 14	**Time :- 2428 **Backtracks :- 15	**Time :- 2595 **Backtracks :- 19

Puzzle 3

Medium2

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

MRV+FC	MRV+FC+AC:	MRV+FC+Prob+AC	MRV+FC+LCV+AC
5 1 9 8 3 6 7 4 2	5 1 9 8 3 6 7 4 2	5 1 9 8 3 6 7 4 2	5 1 9 8 3 6 7 4 2
2 4 6 9 5 7 1 8 3	2 4 6 9 5 7 1 8 3	2 4 6 9 5 7 1 8 3	2 4 6 9 5 7 1 8 3
8 7 3 1 4 2 5 9 6	8 7 3 1 4 2 5 9 6	8 7 3 1 4 2 5 9 6	8 7 3 1 4 2 5 9 6
6 8 7 2 1 4 9 3 5	6 8 7 2 1 4 9 3 5	6 8 7 2 1 4 9 3 5	6 8 7 2 1 4 9 3 5
9 5 4 6 7 3 2 1 8	9 5 4 6 7 3 2 1 8	9 5 4 6 7 3 2 1 8	9 5 4 6 7 3 2 1 8
3 2 1 5 9 8 6 7 4	3 2 1 5 9 8 6 7 4	3 2 1 5 9 8 6 7 4	3 2 1 5 9 8 6 7 4
4 3 5 7 6 1 8 2 9	4 3 5 7 6 1 8 2 9	4 3 5 7 6 1 8 2 9	4 3 5 7 6 1 8 2 9
7 6 8 3 2 9 4 5 1	7 6 8 3 2 9 4 5 1	7 6 8 3 2 9 4 5 1	7 6 8 3 2 9 4 5 1
1 9 2 4 8 5 3 6 7	1 9 2 4 8 5 3 6 7	1 9 2 4 8 5 3 6 7	1 9 2 4 8 5 3 6 7
**Time :- 2351	**Time :- 957	**Time :- 937	**Time :- 944
**Backtracks :- 12	**Backtrack:- 2	**Backtracks :- 2	**Backtracks :- 2

Puzzle 4

Hard1

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

MRV+FC	MRV+FC+AC:	MRV+FC+Prob+AC	MRV+FC+LCV+AC
2 4 8 1 5 6 7 3 9	2 4 8 1 5 6 7 3 9	2 4 8 1 5 6 7 3 9	2 4 8 1 5 6 7 3 9
7 1 3 9 8 4 5 2 6	7 1 3 9 8 4 5 2 6	7 1 3 9 8 4 5 2 6	7 1 3 9 8 4 5 2 6
5 6 9 7 3 2 1 8 4	5 6 9 7 3 2 1 8 4	5 6 9 7 3 2 1 8 4	5 6 9 7 3 2 1 8 4
1 5 6 3 2 8 9 4 7	1 5 6 3 2 8 9 4 7	1 5 6 3 2 8 9 4 7	1 5 6 3 2 8 9 4 7
8 3 7 6 4 9 2 1 5	8 3 7 6 4 9 2 1 5	8 3 7 6 4 9 2 1 5	8 3 7 6 4 9 2 1 5
4 9 2 5 7 1 3 6 8	4 9 2 5 7 1 3 6 8	4 9 2 5 7 1 3 6 8	4 9 2 5 7 1 3 6 8
3 8 5 4 1 7 6 9 2	3 8 5 4 1 7 6 9 2	3 8 5 4 1 7 6 9 2	3 8 5 4 1 7 6 9 2
6 2 1 8 9 5 4 7 3	6 2 1 8 9 5 4 7 3	6 2 1 8 9 5 4 7 3	6 2 1 8 9 5 4 7 3
9 7 4 2 6 3 8 5 1	9 7 4 2 6 3 8 5 1	9 7 4 2 6 3 8 5 1	9 7 4 2 6 3 8 5 1
**Time :- 2664	**Time :- 1907	**Time :- 1593	**Time :- 1350
**Backtracks :- 0	**Backtrack:- 8	**Backtracks :- 8	**Backtracks :- 6

Puzzle 5

Hard2

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

MRV+FC	MRV+FC+AC:	MRV+FC+Prob+AC	MRV+FC+LCV+AC
3 9 5 4 1 8 2 7 6	3 9 5 4 1 8 2 7 6	3 9 5 4 1 8 2 7 6	3 9 5 4 1 8 2 7 6
2 7 6 5 3 9 8 1 4	2 7 6 5 3 9 8 1 4	2 7 6 5 3 9 8 1 4	2 7 6 5 3 9 8 1 4
1 8 4 6 2 7 9 5 3	1 8 4 6 2 7 9 5 3	1 8 4 6 2 7 9 5 3	1 8 4 6 2 7 9 5 3
7 1 8 9 5 6 4 3 2	7 1 8 9 5 6 4 3 2	7 1 8 9 5 6 4 3 2	7 1 8 9 5 6 4 3 2
4 5 3 8 7 2 6 9 1	4 5 3 8 7 2 6 9 1	4 5 3 8 7 2 6 9 1	4 5 3 8 7 2 6 9 1
6 2 9 1 4 3 5 8 7	6 2 9 1 4 3 5 8 7	6 2 9 1 4 3 5 8 7	6 2 9 1 4 3 5 8 7
8 4 2 7 9 1 3 6 5	8 4 2 7 9 1 3 6 5	8 4 2 7 9 1 3 6 5	8 4 2 7 9 1 3 6 5
9 3 7 2 6 5 1 4 8	9 3 7 2 6 5 1 4 8	9 3 7 2 6 5 1 4 8	9 3 7 2 6 5 1 4 8
5 6 1 3 8 4 7 2 9	5 6 1 3 8 4 7 2 9	5 6 1 3 8 4 7 2 9	5 6 1 3 8 4 7 2 9
**Time :- 1986	**Time :- 1509	**Time :- 1381	**Time :- 1301
**Backtracks :- 5	**Backtrack:- 4	**Backtracks :- 5	**Backtracks :- 4

Puzzle 6

Evil1

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

MRV+FC	MRV+FC+AC:	MRV+FC+Prob+AC	MRV+FC+LCV+AC
7 6 9 8 5 3 2 4 1	7 6 9 8 5 3 2 4 1	7 6 9 8 5 3 2 4 1	7 6 9 8 5 3 2 4 1
2 4 3 6 9 1 5 7 8	2 4 3 6 9 1 5 7 8	2 4 3 6 9 1 5 7 8	2 4 3 6 9 1 5 7 8
1 8 5 7 2 4 6 9 3	1 8 5 7 2 4 6 9 3	1 8 5 7 2 4 6 9 3	1 8 5 7 2 4 6 9 3
4 2 1 5 7 8 3 6 9	4 2 1 5 7 8 3 6 9	4 2 1 5 7 8 3 6 9	4 2 1 5 7 8 3 6 9
3 9 6 4 1 2 7 8 5	3 9 6 4 1 2 7 8 5	3 9 6 4 1 2 7 8 5	3 9 6 4 1 2 7 8 5
5 7 8 3 6 9 1 2 4	5 7 8 3 6 9 1 2 4	5 7 8 3 6 9 1 2 4	5 7 8 3 6 9 1 2 4
8 5 2 9 3 7 4 1 6	8 5 2 9 3 7 4 1 6	8 5 2 9 3 7 4 1 6	8 5 2 9 3 7 4 1 6
9 3 7 1 4 6 8 5 2	9 3 7 1 4 6 8 5 2	9 3 7 1 4 6 8 5 2	9 3 7 1 4 6 8 5 2
6 1 4 2 8 5 9 3 7	6 1 4 2 8 5 9 3 7	6 1 4 2 8 5 9 3 7	6 1 4 2 8 5 9 3 7
**Time :- 14105	**Time :- 3392	**Time :- 2375	**Time :- 12547
**Backtracks :- 340	**Backtrack:- 19	**Backtracks :- 14	**Backtracks :- 137

Puzzle 7

Evil2

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

MRV+FC:	MRV+FC+AC:	MRV+FC+Prob+AC	MRV+FC+LCV+AC
8 6 5 4 7 3 1 9 2 9 3 1 5 6 2 8 7 4 4 2 7 8 9 1 5 3 6 3 1 2 7 4 8 9 6 5 7 8 6 9 1 5 4 2 3 5 9 4 2 3 6 7 8 1 1 7 8 6 2 4 3 5 9 2 4 9 3 5 7 6 1 8 6 5 3 1 8 9 2 4 7	8 6 5 4 7 3 1 9 2 9 3 1 5 6 2 8 7 4 4 2 7 8 9 1 5 3 6 3 1 2 7 4 8 9 6 5 7 8 6 9 1 5 4 2 3 5 9 4 2 3 6 7 8 1 1 7 8 6 2 4 3 5 9 2 4 9 3 5 7 6 1 8 6 5 3 1 8 9 2 4 7	8 6 5 4 7 3 1 9 2 9 3 1 5 6 2 8 7 4 4 2 7 8 9 1 5 3 6 3 1 2 7 4 8 9 6 5 7 8 6 9 1 5 4 2 3 5 9 4 2 3 6 7 8 1 1 7 8 6 2 4 3 5 9 2 4 9 3 5 7 6 1 8 6 5 3 1 8 9 2 4 7	8 6 5 4 7 3 1 9 2 9 3 1 5 6 2 8 7 4 4 2 7 8 9 1 5 3 6 3 1 2 7 4 8 9 6 5 7 8 6 9 1 5 4 2 3 5 9 4 2 3 6 7 8 1 1 7 8 6 2 4 3 5 9 2 4 9 3 5 7 6 1 8 6 5 3 1 8 9 2 4 7
**Time :- 11988 **Backtracks :- 287	**Time :- 1776 **Backtrack:- 5	**Time :- 5037 **Backtracks :- 54	**Time :- 1316 **Backtracks :- 4

Puzzle 8

Evil3

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

MRV+FC:	MRV+FC+AC:	MRV+FC+Prob+AC	MRV+FC+LCV+AC
4 2 5 7 1 8 6 9 3 3 9 8 5 4 6 7 2 1 6 7 1 2 9 3 5 4 8 8 3 2 9 6 5 4 1 7 7 5 4 8 2 1 3 6 9 1 6 9 4 3 7 2 8 5 2 4 3 1 7 9 8 5 6 9 8 7 6 5 2 1 3 4 5 1 6 3 8 4 9 7 2	4 2 5 7 1 8 6 9 3 3 9 8 5 4 6 7 2 1 6 7 1 2 9 3 5 4 8 8 3 2 9 6 5 4 1 7 7 5 4 8 2 1 3 6 9 1 6 9 4 3 7 2 8 5 2 4 3 1 7 9 8 5 6 9 8 7 6 5 2 1 3 4 5 1 6 3 8 4 9 7 2	4 2 5 7 1 8 6 9 3 3 9 8 5 4 6 7 2 1 6 7 1 2 9 3 5 4 8 8 3 2 9 6 5 4 1 7 7 5 4 8 2 1 3 6 9 1 6 9 4 3 7 2 8 5 2 4 3 1 7 9 8 5 6 9 8 7 6 5 2 1 3 4 5 1 6 3 8 4 9 7 2	4 2 5 7 1 8 6 9 3 3 9 8 5 4 6 7 2 1 6 7 1 2 9 3 5 4 8 8 3 2 9 6 5 4 1 7 7 5 4 8 2 1 3 6 9 1 6 9 4 3 7 2 8 5 2 4 3 1 7 9 8 5 6 9 8 7 6 5 2 1 3 4 5 1 6 3 8 4 9 7 2
**Time :- 1986 **Backtracks :- 0	**Time :- 4223 **Backtrack:- 27	**Time :- 1778 **Backtracks :- 23	**Time :- 1519 **Backtracks :- 6