Report
Project 2 - Sudoku

Aashray Arora (aashray.arora@stonybrook.edu)
Eshita Bheda (eshita.bheda@stonybrook.edu)
Varun Shastry (varun.shastry@stonybrook.edu)

Question 4.

Solved the problem using the basic Brute Force Backtrack algorithm. It assumes the Solution domain from [1-9] for the 3x3 Sudoku problem. Below is the table of no of nodes expanded and time taken for the five iterations of the code run.

Iteration	Number of nodes expanded	Time taken (in seconds)
1	946	0.040392
2	1225	0.053627
3	454	0.018960
4	1104	0.047252
5	3637	0.160292
Average	1473.2	0.0641046

Variance: 1239330.16

Question 5.

We have added MRV heuristic, Forward checking and Arc consistency to the problem to optimize the solution. This improvises the solution by decreasing the number of nodes expanded.

MRV heuristic: (Minimum Remaining Values)
In this we initially build a solution domain which is a subset of the possible domains formed out by applying the constraints. This chooses the cell with the minimum remaining solutions. Below is the table of no of nodes expanded and time taken for the five iterations of the code run.

Iteration	Number of nodes expanded	Time taken
1	10662	0.442822
2	100	0.006261
3	868	0.039769
4	3547	0.168136
5	3496	0.155430
Average	3734.6	0.1624836

Variance: 13901741.44

Forward Checking - Inference:

This adds the forward checking logic above the MRV algorithm. Forward Checking adds the functionality of ‘refining’ our solution domain everytime we assign a value to (However in the algorithm we do only when the domain contains only one value in it). This adds a great improvisation to the efficiency. The refining is done by removing the variable that is assigned from all the domains which are depended on the current cell/node. Below is the table of no of nodes expanded and time taken for the five iterations of the code run.

Iteration	Number of nodes expanded	Time taken (in seconds)
1	56	0.004568
2	532	0.023425
3	142	0.008246
4	172	0.010300
5	3496	0.157065
Average	879.6	0.0407208

Variance = 1737888.64

Arc Consistency:

This algorithm is an advancement to the Inference made in Forward checking. Forward checking refines only the dependent of the current node/cell while Arc consistency goes ahead and clears all the dependent-of-dependent and so on. This is done only when we encounter a cell with only 1 possible solution. Below is the table of no of nodes expanded and time taken for the five iterations of the code run.

Iteration	Number of nodes expanded	Time taken (in seconds)
1	100	0.006638
2	868	0.040867
3	3496	0.156946
4	552	0.025968
5	56	0.004814
Average	1014.4	0.0470466

Variance: 1629648.64

Conclusion

Forward checking and Arc Consistency adds a great improvisation to the performance and speed of the execution of the algorithm by decreasing the solution domain upon the assignment of a solution. According to the statistics it adds at least a 3-5 fold performance to the backtrack solution.