**Sudoku:**

A Sudoku board with Grid Size ‘N’ has N rows and N columns with a total of N\*N cells. Each cell can be assigned with values from 1 to N.

**Definitions:**

Cell: A cell is a 'square' in a Sudoku grid.

Grid: A grid represents the Sudoku board.

Peers: All the cell's neighbors; neighbors are cells that are in the same unit of the cell.

Unit: A collection of cells, for each row, column and the region which is of size sqrt(N) x sqrt(N) size.

**Sudoku as CSP:**

Variables: Each cell in the Grid ranging from 1 to (N\*N).

Domain: The domain is any digit ranging from 1 to N.

Constraints: The constraints are:

* Same digit can’t appear more than once in the same row.
* Same digit can’t appear more than once in the same column.
* Same digit can’t appear more than once in the same region.

**Forward Checking:**

Forward checking is the method of finding and eliminating the list of possibilities that do match the constraints from the domains of unassigned variables, in advance. Forward checking essentially allows to detect the failure earlier thus resulting in an efficient backtracking and reduced search tree size.

**Implementation:**

Our implementation of Forward Checking uses a HashMap of Cells to List of Possible Assignments for all unassigned cells. Upon assigning a domain value to a Cell, the map is consulted for detecting any unassigned cell that will run out of possible assignments and does backtracking if there is one. Also, constraints are re-evaluated for the cell and its peers and the map is updated with that information. This allows to detect an upcoming failure and try different possible assignments to mitigate it.