Sudoku Solver using AI

Made by: Anurag Shah, B16CS034

Chinmay Garg, B16CS041

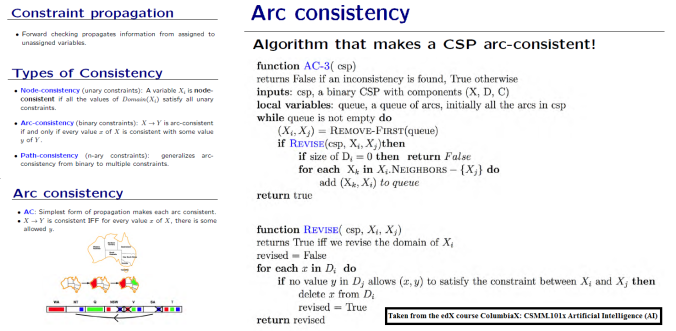
Sudoku is a logic-based combinatorial number placement puzzle.

Given a partially filled 9×9 2D array ‘grid[9][9]’, the goal is to assign digits (from 1 to 9) to the empty cells so that every row, column, and sub-grid of size 3×3 contains exactly one instance of the digits from 1 to 9.

Algorithms used:

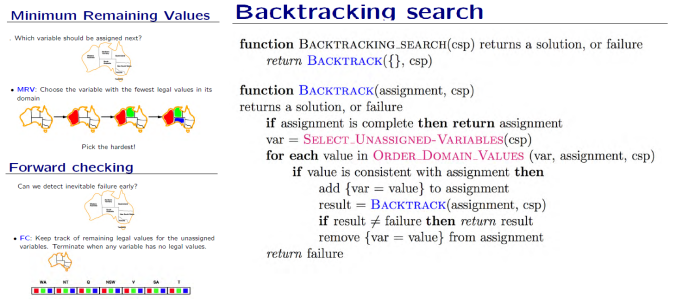
1. AC-3 Algorithm

First, the AC-3 (arc-consistency checking) algorithm will be implemented. This algorithm will propagate the constraints and reduce the domain size of the variables by ensuring all possible (future) assignments consistent.



1. Backtracking Algorithm

Like all other Backtracking problems, we can solve Sudoku by one by one assigning numbers to empty cells. Before assigning a number, we check whether it is safe to assign. We basically check that the same number is not present in the current row, current column and current 3X3 sub-grid. After checking for safety, we assign the number, and recursively check whether this assignment leads to a solution or not. If the assignment doesn’t lead to a solution, then we try next number for the current empty cell. And if none of the number (1 to 9) leads to a solution, we return false.



STEPS:

1. Every cell has a corresponding vector associated to it (2-dimensional vector named ‘range’).
2. We store all the possible assignments to a cell in it.
3. We check if the first value in the ‘range’ of the present cell is consistent/valid or not.
4. If the value is valid, then we temporarily remove that ‘value’ from the ‘range’ of all other cells in the same row, column and sub-grid.
5. Then we move to the next cell and repeat from step 3.
6. If ‘range’ of some cell is empty, we go back to the previous cell and add the removed ‘value’ back to the cells in same row, column and sub-grid. The ‘value’ is removed from the present cell.
7. Repeat from step 3.
8. If we reach the last cell, then the sudoku is solved.