CSE537 Artificial Intelligence
Project 3 – Sudoku

Team Details

Name	**SBU ID**	**Net ID**
Shahzeb Nihalahmed Patel	110369918	shahpatel
Rahool Paliwal	110570213	rpaliwal

Input:

12,3,4;

-,-,1,-,12,9,-,8,-,-,6,-;

10,8,12,-,-,-,-,-,-,1,2,-;

-,3,-,-,-,2,-,-,-,-,12,10;

-,-,-,1,-,-,5,4,2,-,-,-;

12,-,-,5,11,-,-,10,-,-,-,8;

-,-,-,2,-,8,9,-,-,5,-,11;

8,-,9,-,-,10,3,-,1,-,-,-;

5,-,-,-,1,-,-,2,6,-,-,9;

-,-,-,10,9,6,-,-,7,-,-,-;

9,11,-,-,-,-,7,-,-,-,10,-;

-,6,10,-,-,-,-,-,-,8,11,5;

-,5,-,-,10,-,11,9,-,4,-,-;

Output + Application

Backtracking:

Execution Time: 6.560327

Consistency Checks: 3099977

Solution: (True, [[7, 2, 1, 11, 12, 9, 10, 8, 5, 3, 6, 4], [10, 8, 12, 9, 5, 4, 6, 3, 11, 1, 2, 7], [6, 3, 5, 4, 7, 2, 1, 11, 8, 9, 12, 10], [11, 10, 8, 1, 3, 7, 5, 4, 2, 6, 9, 12], [12, 9, 6, 5, 11, 1, 2, 10, 4, 7, 3, 8], [4, 7, 3, 2, 6, 8, 9, 12, 10, 5, 1, 11], [8, 12, 9, 6, 4, 10, 3, 7, 1, 11, 5, 2], [5, 4, 7, 3, 1, 11, 12, 2, 6, 10, 8, 9], [2, 1, 11, 10, 9, 6, 8, 5, 7, 12, 4, 3], [9, 11, 4, 12, 8, 5, 7, 6, 3, 2, 10, 1], [3, 6, 10, 7, 2, 12, 4, 1, 9, 8, 11, 5], [1, 5, 2, 8, 10, 3, 11, 9, 12, 4, 7, 6]])

Backtracking can be applied only for problems which admit the concept of a "partial candidate solution" and a relatively quick test of whether it can possibly be completed to a valid solution. It is useless, for example, for locating a given value in an unordered table. When it is applicable, however, backtracking is often much faster than brute force enumeration of all complete candidates, since it can eliminate a large number of candidates with a single test.

backtrackingMRV:

Execution Time: 6.331595

Consistency Checks: 42005

Solution: (True, [[7, 2, 1, 11, 12, 9, 10, 8, 5, 3, 6, 4], [10, 8, 12, 9, 5, 4, 6, 3, 11, 1, 2, 7], [6, 3, 5, 4, 7, 2, 1, 11, 8, 9, 12, 10], [11, 10, 8, 1, 3, 7, 5, 4, 2, 6, 9, 12], [12, 9, 6, 5, 11, 1, 2, 10, 4, 7, 3, 8], [4, 7, 3, 2, 6, 8, 9, 12, 10, 5, 1, 11], [8, 12, 9, 6, 4, 10, 3, 7, 1, 11, 5, 2], [5, 4, 7, 3, 1, 11, 12, 2, 6, 10, 8, 9], [2, 1, 11, 10, 9, 6, 8, 5, 7, 12, 4, 3], [9, 11, 4, 12, 8, 5, 7, 6, 3, 2, 10, 1], [3, 6, 10, 7, 2, 12, 4, 1, 9, 8, 11, 5], [1, 5, 2, 8, 10, 3, 11, 9, 12, 4, 7, 6]])

MRV heuristics picks a variable that is most likely to cause a failure soon, thereby pruning the search tree. If there is a variable X with zero legal values remaining, the MRV heuristic will select X and failure will be detected immediately—avoiding pointless searches through other variables which always will fail when X is finally selected.

backtrackingMRVfwd:

Execution Time: 6.384716

Consistency Checks: 34245

Solution: (True, [[7, 2, 1, 11, 12, 9, 10, 8, 5, 3, 6, 4], [10, 8, 12, 9, 5, 4, 6, 3, 11, 1, 2, 7], [6, 3, 5, 4, 7, 2, 1, 11, 8, 9, 12, 10], [11, 10, 8, 1, 3, 7, 5, 4, 2, 6, 9, 12], [12, 9, 6, 5, 11, 1, 2, 10, 4, 7, 3, 8], [4, 7, 3, 2, 6, 8, 9, 12, 10, 5, 1, 11], [8, 12, 9, 6, 4, 10, 3, 7, 1, 11, 5, 2], [5, 4, 7, 3, 1, 11, 12, 2, 6, 10, 8, 9], [2, 1, 11, 10, 9, 6, 8, 5, 7, 12, 4, 3], [9, 11, 4, 12, 8, 5, 7, 6, 3, 2, 10, 1], [3, 6, 10, 7, 2, 12, 4, 1, 9, 8, 11, 5], [1, 5, 2, 8, 10, 3, 11, 9, 12, 4, 7, 6]])

The choice of the next variable to evaluate is particularly important, as it may produce exponential differences in running time. In order to prove unsatisfiability as quickly as possible, variables leaving few alternatives after assigned are the preferred ones. This idea can be implemented by checking only satisfiability or unsatisfiability of variable/value pairs. In particular, the next variable that is chosen is the one having a minimal number of values that are consistent with the current partial solution

backtrackingMRVcp:

Execution Time: 6.058935

Consistency Checks: 2152

Solution: (True, [[7, 2, 1, 11, 12, 9, 10, 8, 5, 3, 6, 4], [10, 8, 12, 9, 5, 4, 6, 3, 11, 1, 2, 7], [6, 3, 5, 4, 7, 2, 1, 11, 8, 9, 12, 10], [11, 10, 8, 1, 3, 7, 5, 4, 2, 6, 9, 12], [12, 9, 6, 5, 11, 1, 2, 10, 4, 7, 3, 8], [4, 7, 3, 2, 6, 8, 9, 12, 10, 5, 1, 11], [8, 12, 9, 6, 4, 10, 3, 7, 1, 11, 5, 2], [5, 4, 7, 3, 1, 11, 12, 2, 6, 10, 8, 9], [2, 1, 11, 10, 9, 6, 8, 5, 7, 12, 4, 3], [9, 11, 4, 12, 8, 5, 7, 6, 3, 2, 10, 1], [3, 6, 10, 7, 2, 12, 4, 1, 9, 8, 11, 5], [1, 5, 2, 8, 10, 3, 11, 9, 12, 4, 7, 6]])

Constraint propagation enumerate the possible values a set of variables may take. Informally, a finite domain is a finite set of arbitrary elements. A constraint satisfaction problem on such domain contains a set of variables whose values can only be taken from the domain, and a set of constraints, each constraint specifying the allowed values for a group of variables. A solution to this problem is an evaluation of the variables that satisfies all constraints. In other words, a solution is a way for assigning a value to each variable in such a way that all constraints are satisfied by these values.

minConflict:

Execution Time: 13.753612

Consistency Checks: 10000

Solution: (False, [[0, 0, 1, 0, 12, 9, 0, 8, 0, 0, 6, 0], [10, 8, 12, 0, 0, 0, 0, 0, 0, 1, 2, 0], [0, 3, 0, 0, 0, 2, 0, 0, 0, 0, 12, 10], [0, 0, 0, 1, 0, 0, 5, 4, 2, 0, 0, 0], [12, 0, 0, 5, 11, 0, 0, 10, 0, 0, 0, 8], [0, 0, 0, 2, 0, 8, 9, 0, 0, 5, 0, 11], [8, 0, 9, 0, 0, 10, 3, 0, 1, 0, 0, 0], [5, 0, 0, 0, 1, 0, 0, 2, 6, 0, 0, 9], [0, 0, 0, 10, 9, 6, 0, 0, 7, 0, 0, 0], [9, 11, 0, 0, 0, 0, 7, 0, 0, 0, 10, 0], [0, 6, 10, 0, 0, 0, 0, 0, 0, 8, 11, 5], [0, 5, 0, 0, 10, 0, 11, 9, 0, 4, 0, 0]])

(Note: the above approach of random nodes generation failed for 12 by 12 marks. But we are expecting credits for our approach on implementation of minConflict algorithm)

CSP algorithm randomly selects a variable from the set of variables with conflicts violating one or more constraints of the CSP.[1] Then it assigns to this variable the value with that minimizes the number of conflicts. If there is more than one value with a minimum number of conflicts, it chooses one randomly. This process of random variable selection and min-conflict value assignment is iterated until a solution is found or a pre-selected maximum number of iterations is reached