Assignment 2: Constraint Satisfaction
Due Date: 11.59 PM 10/19/2022

1. **(15 points)** In the following sub-parts, "briefly describe" means provide a description of a problem in a couple of sentences such that the constraints are clear, and identify the variables, their domains, and what aspect of the problem they correspond to. For example, if you were to "briefly describe" the map coloring problem from the slides, an acceptable answer would be, "Color the regions of a map with different colors, drawn from a fixed, finite set of colors, such that neighboring (touching) regions do not have the same color. Each region is a finite variable, whose domain is the set of available colors."

   (a) Briefly describe a CSP where the variables have a finite domain. You may not use map coloring.

   (b) Briefly describe a CSP where the variables have an infinite, but discrete, domain.

   (c) Briefly describe a CSP where the variables have a continuous domain.

2. **(15 points)** Consider the constraint graph of Figure 1 with named binary constraints. $r1$ is a relation on A and B, which we can write as $r1(A, B)$, and similarly for the other relations. Consider solving this network using Variable Elimination Algorithm.

   (a) Suppose you were to eliminate variable E. Which constraints are removed? New constraints are created on which variables?

3. **(70 points) Programming Assignment:**
   In class, we discussed Sudoku puzzles as an example of Constraint Satisfaction Problems (CSPs) in AI. In a CSP, the goal is to find a complete, consistent assignment of values to a set of variables X (taken from their domains D) satisfying a set of constraints C that limit the valid combinations of variable values.

   As a reminder, a Sudoku puzzle is a 9x9 grid (81 variables) where each cell in the grid can take on the integer values 1-9 (the domain of each variable). A solution to a Sudoku puzzle is an assignment of values for each cell in the grid such that no two cells in the same row, column, or 3x3 square have the same value.

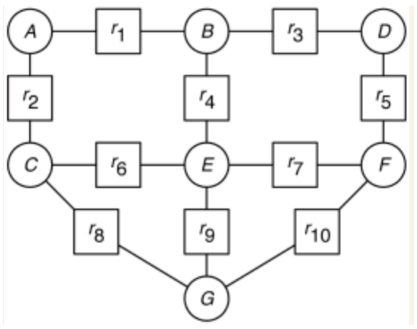

Figure 1

```
..3|.2.|6..
9..|3.5|..1
..1|8.6|4..
-----------
..8|1.2|9..
7..|...|..8
..6|7.8|2..
-----------
..2|6.9|5..
8..|2.3|..9
..5|.1.|3..
```

(a) Unsolved Sudoku Grid 1

```
483|921|657
967|345|821
251|876|493
-----------
548|132|976
729|564|138
136|798|245
-----------
372|689|514
814|253|769
695|417|382
```

(b) Solved Sudoku Grid 1

Figure 2

```
4 3 5 | 2 6 9 | 7 8 1
6 8 2 | 5 7 1 | 4 9 3
1 9 7 | 8 3 4 | 5 6 2
-----------------------
8 2 6 | 1 9 5 | 3 4 7
3 7 4 | 6 8 2 | 9 1 5
9 5 1 | 7 4 3 | 6 2 8
-----------------------
5 1 9 | 3 2 6 | 8 7 4
2 4 8 | 9 5 7 | 1 3 6
7 6 3 | 4 1 8 | 2 5 9
```

Figure 3: Solved Sudoku Grid

In this assignment, you have to model a unsolved **Sudoku puzzle as CSP** and then solve it using the **Constraint Propagation** (AC3 algorithm) and **Backtracking Search**. In AC3, you need to check arc consistency and enforce arc consistency by removing values from the domain of unassigned variables.

Note: You are free to develop your program and represent your data structures however you wish.

Extra Credit (20 points): Use **Most Constrained Variable** and **Least Constraining Value** to choose next variable and value to assign.

(a) **Input:** Your program should take an unsolved Sudoku Grid as input. The input will be given as a continuous string of 81 characters. The unassigned cells will be marked with a **"."** The input string is the join of consecutive rows.

For example, for the unsolved grid in Figure 2a. The input to your program would be "..3.2.6..9..3.5..1..18.64....81.29..7.......8..67.82....26.95..8..2.3..9..5.1.3.."

(b) **Output:** Your program should print the final output as a 9x9 grid. It should look something like Figure 2b.

(c) **Testing:** To test your program, you may use the following inputs:

i. "..3.2.6..9..3.5..1..18.64....81.29..7.......8..67.82....26.95..8..2.3..9..5.1.3.."

Solution should be Figure 2b.

ii. "...26.7.168..7..9.19...45..82.1...4...46.29...5...3.28..93...74.4..5..367.3.18..."
Solution should be Figure 3.

(d) **What to submit:** Your code should be well commented and easy to read. Submit the code along with relevant instructions to run the code. If you are using a **non python** coding language, check with your TA whether you need to provide a makefile.

Additionally, add a pdf or word file to your package and provide answers to the following questions:

i. A CSP has a variable and each variable has a domain. For the Sudoku puzzle, what are the variable and what are their domains?

ii. What data structure did you use to store your Sudoku Puzzle?

iii. What constraints did you add for the puzzle? Provide snippet of your code where you define the constraints.

iv. To implement AC3 you need to:

A. Check for Arc Consistency

B. Implement Arc Consistency

Describe how you implemented this and provide snippets of your code to support your answer.

v. **Extra Credit:** If you use MCV and LRV, describe how you implemented this and provide snippets of your code to support your answer.