Hamin Han

Constraint Satisfaction

1. 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.

a) Briefly describe a CSP where the variables have a finite domain.

The 8-Queens problem. On a 8x8 grid, place 8 queens in each row so that no queen is attacking each other, so no queens on the same column and no queens on the same diagonal. Each queen is a variable, whose domains is the 8 columns of the board.

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

Scheduling construction jobs onto a calendar, each job's start date is a variable and the possible values are integer numbers of days from the current date.

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

"For example, the scheduling of experiments on the Hubble Space Telescope requires very precise timing of observations; the start and finish of each observation and maneuver are continuous-valued variables that must obey a variety of astronomical, precedence, and power constraints." (http://aima.cs.berkeley.edu/newchap05.pdf)

2. 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?

R4,r6,r6,r9 is gone. There would be new constraints between B and G as well as C and F.

3. Programming Assignment 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?

Variables are the “cells” or “tiles” in the Sudoku grid. Their domains are 1-9.

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

The puzzle itself was a list of list. Sometimes it was made into a singular list too. As for the domains, constraints, etc, dictionaries and lists were used.

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

```
# Constraints
# Binary constraints
self.constraints = [(variable, neighbor) for variable in self.variables for neighbor in self.neighbors[variable]]
```

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.

```python
# AC3 algorithm for Constraint Propagation
def AC3(self):
    queue = self.constraints

    while queue:

        (x, y) = queue.pop(0)

        if self.revise(x, y):
            if len(self.domains[x]) == 0:
                return False
            for i in self.neighbors[x]:
                if i != x:
                    queue.append((i,x))

    return True

# Revise algorithm used in AC3
def revise(self, x, y):

    revised = False

    for i in self.domains[x]:

        if not any([(i != j) for j in self.domains[y]]):

            # Remove i from x's domain
            self.domains[x].remove(i)
            revised = True

    return revised
```

Similar to the example on the slides with the stack of all variables and removing arc inconsistencies. Within the while loop, check if consistent, if not reduce the domain. And If it was changed, add it back to the queue.