Spring Semester, 2008 B.Y. Choueiry

CSCE 421/821: Foundations of Constraint Processing

Homework 5

Assigned: Wednesday, April 9, 2008

Due: Wednesday, April 16, 2008

Total value: 100 points + 30 points bonus.

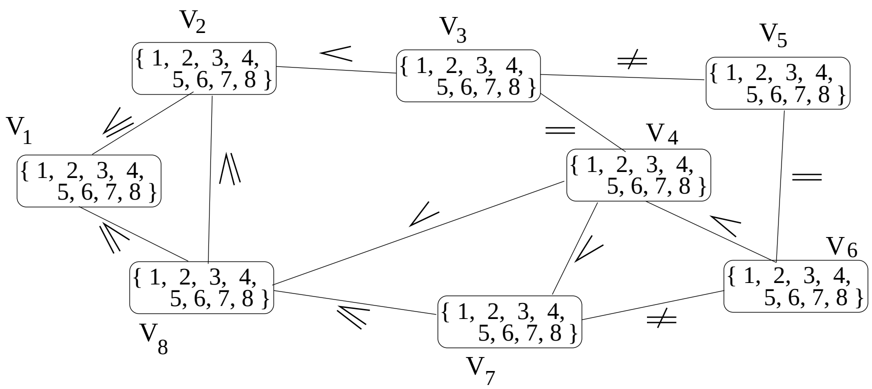
Notes: This homework must be done individually. *If you receive help from anyone, you must clearly acknowledge it.* Always acknowledge sources of information (URL, book, class notes, etc.). Please inform instructor quickly about typos or other errors.

Contents

1 (8 points) Temporal network

Courtesy of Rina Dechter

Consider the following constraint satisfaction problem with 8 variables.



1. Show the graph of the equivalent arc-consistent problem. (4 points)

2. Show the graph of the equivalent path-consistent problem. (4 points)
   Don't forget to generate the complete network.

2 (20 points) A simple application of path-consistency

Consider a CSP with the three variables x, y, and z, each with the domain $\{1, 2, 3, 4, 5\}$, and the following three symmetric constraints:

$$C_{x,y} = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 \end{bmatrix} \quad C_{y,z} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad C_{x,z} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

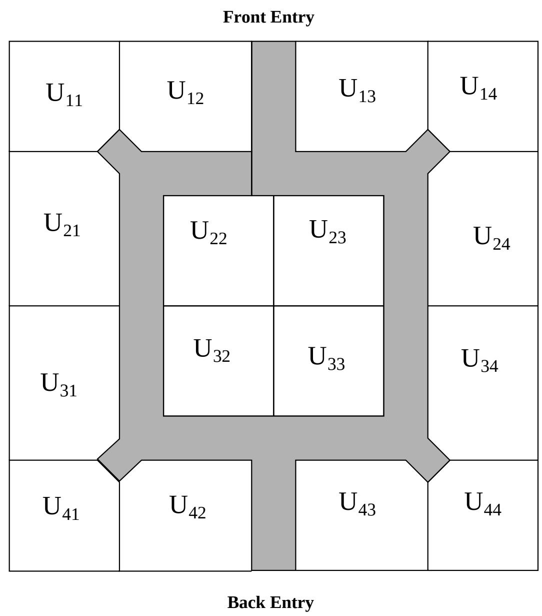
Define the domains of the variables as vectors. Give the six constraints (C_x, C_y, C_z, $C_{x,y}$, $C_{y,z}$, $C_{x,z}$) that result from applying `PC-1` to the CSP. Follow `PC-1` as closely as possible and document the application of each constraint-filtering operation you apply.

3 (6 points) Tree graph

Consider a constraint graph structured as a tree, any tree. Apply the procedure for finding the width of this graph (will be discussed in class).

1. **(3 points)** What is the width of this graph?

2. **(3 points)** State and demonstrate a theorem about the width of a tree-structured graph.



4 (10 points + bonus 5 points) Vacation house

The diagram below is the layout grid for a group vacation-hall. Each square in the grid is a unit. If a room takes up more than two units, the two units assigned must be adjacent (i.e., share a wall). You need to assign the functionality to each unit so that the architect can finish the details. There are sixteen units in the grid. The shaded area represents the halls. Use the following information to help you model the CSP.

- The kitchen takes up two units and should be at the far back of the house.
- The dining hall takes up two units and must be next to the kitchen.
- There are six bedrooms, each of one unit. The bedrooms must be clustered.
- The billard room (1 unit) must be next to the lounge (2 units).
- The Library (1 unit) must be at the front of the building and must have a window.
- Half of the bedrooms must have windows.
- The billard room and exercise room (each of 1 unit) should *not* be next to the bedrooms and should both have windows.
- The office (one unit) must be adjacent to the front hall entry.

1. **(3 points)** Give the variables and their domains.

2. **(3 points)** What are the assumptions that you must make to complete the model, there are at least five. Discuss three of them.

3. **(2 points)** Give, in extension, the most restrictive unary constraint in your model (i.e., this is the constraint that most reduces the domain of the variable to which it applies).

4. **(2 points)** Choose a binary constraint and give its definition in intension, using set notation and proper constraint labeling. When more than one binary constraints exist between two given nodes, you must provide the resulting constraint that satisfies them.

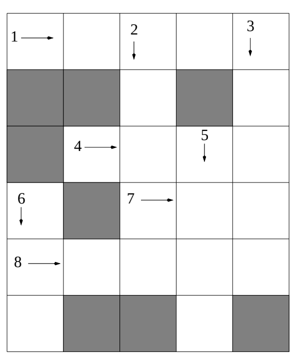
5. **(Bonus 5 points)** Discuss how unary constraints can be combined with binary constraints to tighten the binary constraints at the preprocessing stage, before search. Give an example of this situation with a unary constraint and a binary constraint in your model, showing, in extension, the effect on the binary constraint both before and after filtering.

5 (31 points) Crossword Puzzle

Consider the list of words:

aft	laser
ale	lee
eel	line
heel	sails
hike	sheet
hoses	steer
keel	tie
knot	

for the crossword puzzle:



The numbers 1, 2, . . ., 8 in the crossword puzzle correspond to the words that will start at those locations. The arrows correspond to the direction of the alignment of the letters in the words. Each word appears at most in the puzzle.

1. **(18 points)** Model this puzzle as a binary CSP. State what are the variables, their respective domains, and the binary constraints between variables.
   Hint: model every variable as a vector (i.e., one dimension array) and express the constraints as an equality between the respective positions of two arrays.

2. **(1 points)** Draw the constraint graph.

3. **(10 points)** Make this CSP arc-consistent.

4. **(2 points)** Give the size of the CSP before *and* after enforcing arc-consistency.

6 (25 points + bonus 25 points) A crypto-arithmetic puzzle

Consider the crypto-arithmetic puzzle shown below.

- You are asked to replace each letter by a *different* digit from 0 to 9.
- No leading zeros are allowed.
- When each letter is replaced by the appropriate number, this cryptogram represents a correct addition problem:

$$\begin{array}{ccccc} & \mathbf{S} & \mathbf{E} & \mathbf{N} & \mathbf{D} \\ + & \mathbf{M} & \mathbf{O} & \mathbf{R} & \mathbf{E} \\ \hline \mathbf{M} & \mathbf{O} & \mathbf{N} & \mathbf{E} & \mathbf{Y} \end{array}$$

1. *Model this puzzle as a CSP.* (20 points)

   List the variables, their domains, and the constraints. Specify the constraint definitions in intension. [Hints: (1) In addition to each letter being a variable, you need to account for the carries. (2) The CSP is not necessarily binary.]

2. *Draw the constraint network of your model.* (5 points)

   Label the nodes with the domains of the variables (in extension), and the constraints with their definitions (in intension).

3. *Solve the puzzle.* (Bonus: 25 points)

   Depending on how you model the puzzle, you may be able to solve it with simple consistency checking on likely non-binary constraints (e.g., node, relational arc-consistency, and relational path-consistency) or may need to simulate a backtrack search. The former may be useful when your model has several constraints of arity 5 or less, the latter may be necessary when your model has one constraint of large arity. (We advise you to adopt the former approach.)

- If your model lends itself to consistency checking, then step through a consistency checking process showing which constraint is checked at each step and how the domains of the applicable variables are updated. Keep in mind that you do not need to visit all the constraints once before you can 'come back' to a given constraint.
- Otherwise, show how you can solve this problem with a backtrack search with a full look-ahead technique (i.e., applying the look-ahead strategy known as Maintaining Arc-Consistency, MAC).

Relational consistency

It may be useful to understand the relational-consistency methods in order to solve the crypto-arithmetic puzzle using constraint propagation only (i.e., without search). Refer to Chapter 8 of Dechter's textbook to learn the details of advanced consistency concepts and methods. In class we discussed arc-consistency and path-consistency in great detail in the context of binary constraints. In the context of non-binary constraints, these concepts need to be generalized to relational arc-consistency (a.k.a. relational 1-consistency), relational path-consistency (a.k.a. relational 2-consistency), and Generalized Arc-Consistence (a.k.a. GAC and relational (1,1)consistency). Below is a summary of the class discussion.

The goal of arc consistency in binary CSPs is to update the domain of each of the variables to which it applies. For non-binary constraints, Generalized Arc-Consistency can be achieved by direct application of the Waltz algorithm (see Lecture slides 4). The two procedures of the Waltz algorithms `Revise` and `Refine` ensure that, for a given constraint, every value of the domain of every variable is supported according to the constraint by some values in the domains of the remaining variables. For example, in Fig 1, if the constraint C_x is defined such as $V_1 + 2V_2 + V + 3 = V_4$, then we consider all combinations of tuples for these variables, remove the combinations that do not satisfy the constraints, then remove from the domains of the variables those values that do not appear in any acceptable combination.

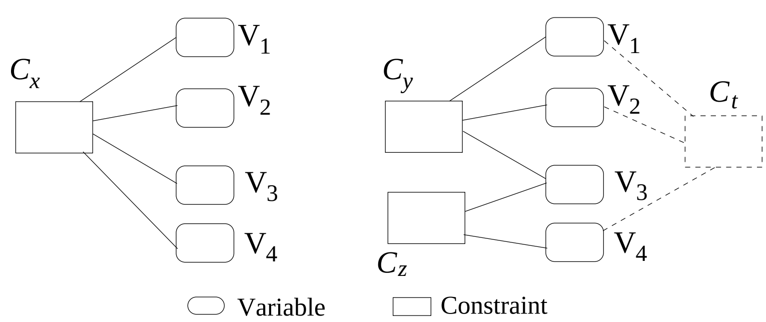


Figure 1: *Left:* Generalized Arc-Consistency. *Right:* Relational path-consistency.

Another extension of arc consistency to non-binary constraints can be seen as guaranteeing that any *consistent* partial solution to all but one of the variables in the scope of the constraint can be extended to this last variable. This property is called relational 1-consistency or relational arc-consistency.

The goal of path consistency for binary constraints is to combine two constraints C_1 an C_2 and induce a new constraint C_3 between the variables that are not in the scope of both C_1 an C_2. The generalization of the path consistency in binary CSPs to relational path-consistency in non-binary CSPs (a.k.a. relational 2-consistency) requires, for two non-binary constraints C_y an C_z, every tuple of length equal (|scope(C_y) $\cup$ scope(C_z)| -1) that is consistent with C_y and C_z, simultaneously, can be extended to the last variable in the scope(C_y) $\cup$ scope(C_z). Generally speaking, enforcing this consistency property may be require adding constraints of arity (|scope(C_y) $\cup$ scope(C_z)| -1).

In the example of Figure 1, the scope of C_t is $\{V_1, V_2, V_4\}$. In the case of algebraic expression, this can be achieved by simple elimination. For example, the two constraints:

$$C_y : \quad V_1 + V_2 \quad > V_3 \tag{1}$$

$$C_z : \quad V_3 + 10 \quad > V_4 \tag{2}$$

yield the following induced constraint

$$C_t : \quad V_1 + V_2 + 10 \quad > V_4. \tag{3}$$