Fall 2003 **B.Y. Choueiry**

CSCE 421/821: Foundations of Constraint Processing

Homework 2

Assigned: Friday, Sep 26, 2003

Due: Friday, Oct 10, 2003, 10:30 am (at the start of class)
Exercises 1 and 2 of homework 2 must be printed out and physically handed to the instructor.
Exercise 3 of homework 2 must be electronically submitted using Webhandin, assignment 2.

Total value: 108 points

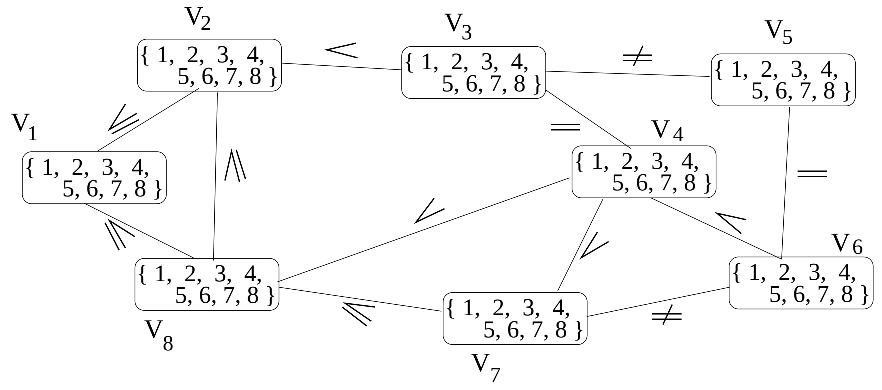
Notes: This homework must be done individually. Please avoid discussing it with classmates, TA or instructor. *If you receive help from anyone, you must clearly acknowledge it.* Always acknowledge sources of information (URL, book, class notes, etc.). Please inform instructor quickly about typos or other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

---

1. Temporal network **(Total 20 points)**

Courtesy of Rina Dechter

Consider the following constraint satisfaction problem with 8 variables. Find the equivalent arc- and path-consistent problem. (each 10 points)



2. **Relational algebra** **(Total 8 points)**

Given the three variables X_1, X_2, and X_3, each with the domain $\{1, 2, 3, 4, 5\}$, and the three relations shown in the tables below:

R_{X_1,X_2}

x_1	x_2
1	2
1	3
2	3
2	4
2	5
3	1
3	2
3	4
4	3
4	5
5	1
5	2

R_{X_2,X_3}

x_2	x_3
1	2
1	3
2	3
2	4
2	5

R_{X_1,X_3}

x_1	x_3
1	1
1	2
1	3
2	4
2	5
3	2
3	3
3	4
5	4
5	5

Compute or do the following:

(a) $R'_{X_1,X_3} = \pi_{X_1,X_3}(R_{X_1,X_2} \bowtie R_{X_2,X_3})$ (2 point)

(b) $R''_{X_1,X_3} = R_{X_1,X_3} \cap R'_{X_1,X_3}$ (2 point)

(c) Using R_{X_1,X_2}, R_{X_2,X_3}, and R''_{X_1,X_3} as the constraints of a CSP, perform Arc Consistency on the problem. What are the final domains? (4 points)

3. **Race problem** **(Total 80 points)**

The following problem is a programming assignment. It is broken down into several sections. In order to demonstrate the work or function of each section, you are required to implement a menu system that allows the grader to enter number and any parameters (if needed) to print out the results to screen. You will be told what menu prompt is required for each section. The menu need be only a simple command line prompt system. A demonstration will be given in class of a sample menu system to help clarify what is required. Consider the following puzzle, known as the *race problem.*

Formulation: There are five competitors, each from a different *university*, each wearing a t-shirt of different *color*, each carrying a different *good-luck charm* (one carries a horse show charm), each preferring a different *sports drink* (one just loves gatorade), and each indulging in a different *victory celebration* when he wins a race. Consider the following facts:

(a) The competitor from Kansas University carries the rabbits foot for good luck.

(b) The competitor who guzzles ‘Recharge’ before the race represents Iowa State University.

(c) The racer decked out in a white T-shirt crossed the finish line immediately in front of the one in the green T-shirt.

(d) The competitor from Nebraska (UNL) is wearing the red T-shirt.

(e) The competitor that take a champange shower when he wins a race carries a saint’s medal for luck.

(f) The racer in the yellow T-shirt performs hand-springs after he has won a race.

(g) The competitor who drinks ‘Energize’ placed in the center of the pack.

(h) The race was won by the competitor from Oklahoma State.

(i) The competitor who performs a victory dance when he wins placed either right before or right after the racer who wears his lucky socks in every race.

(j) The competitor that performs hand-springs when he wins a race placed either immediately before of immediately after the racer that carries a shiny new penny to improve his luck.

(k) The competitor who wears the green shirt drinks ‘Powerade’ during the race.

(l) The racer who tries to kiss as many of fan as possible after he wins really believes in drinking ‘Quencher’ during the race.

(m) The competitor from Texas A&M gives his coach (his wife) a big hug when he wins a race.

(n) The racer wearing the blue shirt placed either immediately before or immediately after the racer from Oklahoma State.

Query: Find out where each competitor studies, what color his T-shirt was for the race, what good-luck charm he carries, what sports drink he prefers and what victory ritual he indulges in when he wins.

Our goal is to model this problem as a CSP, then test some consistency checking algorithms on this CSP. In the following homework, you will have to encode backtrack search algorithms for this CSP.

(a) **(20 points)** Articulate the variables, the values and the constraints in extension. There is no menu item needed for this part in your implementation. However, a README.txt file containing the variable names, initial domains, and constraints in extension must be electronically submitted with your code.

- Make a variable for each university, t-shirt color, drink, *etc.*, so that you have 25 variables.
- The domain of each variable should be *finish position*: {1, 2, 3, 4, 5}. For example, assigning the value 2 to the variable new penny carrier means that the new penny carrier came in second place.

- Express all the above enumerated conditions as either unary or binary constraints.

(b) **(10 points)** Implement, in the language of your choice, the data structures for representing the variables, their domain and the constraints. Print out the CSP variables and domains. Each menu item that is required is listed with the description of its function

Menu Item 1: "Show variables and domain for each variable" result - print out to screen of each variable and associated domain, then redisplays menu.

Menu Item 2: "Show Constraints"
result - print out to screen all constraint in extension, then redisplays menu.

(c) **(5 points)** Write a function that takes a variable-value pair and a constraint and checks whether the VVP is supported by the other variable in the (binary) constraint.

Menu Item 3: "Variable-Value pair"
result - prompt for variable name, value, and constraint
result - returns 'yes' if variable is supported, 'no' if not, then redisplays menu.

(d) **(5 points)** Write a function that takes a variable and returns its 'neighboring' variables, i.e. variables that share a constraint with it.

Menu Item 4: "Neighbors"
result - prompt for variable name
result - list of neighbors printed to screen and redisplays menu.

(e) **(2 points)** Write a function that determines the degree of a variables.

Menu Item 5: "Variable degree"
result - prompt for variable name.
result - prints degree to screen, and redisplays menu.

(f) **(10 points)** Implement and run node consistency.

- Print out and turn in the resulting CSP's variables and domains.
- Report the number of constraint checks performed. A constraint count is incremented each time two VVP's are checked for consistency. With 2 VVPs and a constraint such as $VVP(V_1, 5), VVP(V_2, 4), C_{1,2}$, you are checking if value 5 for V_1 and value 4 for V_2 are allowed in the constraint $C_{1,2}$. This is one constraint count.
- What is the size of the resulting CSP?

Menu Item 6: "Run node consistency"
result - Print variables with resulting domains to screen along with constraint check count, then redisplay menu.

(g) **(20 points)** Implement and run arc-consistency (AC-3).

- Print out and turn in the resulting CSP's variables and domains.
- Report the number of constraint checks performed.
- What is the size of the resulting CSP?

Menu Item 7: "Run arc consistency AC-3"
result - Print variables with resulting domains to screen along with constraint check count, then redisplay menu.

(h) Write functions that sort variables by:
   i. **(3 points)** increasing domain size
   ii. **(2 points)** increasing node degree
   iii. **(3 points)** increasing ratio of domain size over degree

These heuristics break ties lexicographically and will be used for ordering heuristics during search in the following homework.

Menu Item 8: "Show ordering by increasing domain size"
result - Print variables in new order with resulting domain size (along with domains) to screen, then redisplay menu.

Menu Item 9: "show ordering by node degree"
result - Print variables in order of increasing node degree (along with domains) to screen, then redisplay menu.

Menu Item 10: "show ordering by increasing ratio of domain size over degree"
result - Print variables in order of increasing ratio (along with domains) to screen, then redisplay menu.

General indications:

- *Please make sure that you keep your code and protect your files.* Your name, date, and course number must appear in each file of code that you submit.
- Declare a data structure to store a given instance of a CSP. For example, this can be a record with the following attributes: a unique problem identifier, a list (or hash-table) of the variables in the CSP, a list (or hash-table) of the constraints in the CSP.
- Declare data structures to represent the CSP variables and constraints. For instance, the data structure for representing a CSP variable can be a record that has a unique identifier, a name of the variable, its initial domain, its current domain (filtered), its assigned value, a set of pointers to the constraints that apply to it. The data structure for the constraint can be a record that has a unique identifier, a set of pointers to the variables in the scope of the constraint, and constraint definition that list all variable-value pairs that are consistent (constraint defined by extension), and/or a method that determined whether two variable-value pairs are consistent (constraint defined in intension).
- Before you test it on the Race problem, test and debug your algorithm on small, manageable examples. Consider the n-queen problem or the scheduling example in Pages 30-31 of Dechter's book.