Homework 3

Assigned: Monday, October 13, 2003.

Due: Monday, November 3, 2003, *by 10:30 a.m.*
Exercise 1, 2, 3 should be printed out and handed to the instructor
Code for exercise 4 should be submitted via Webhandin.

Total value: 100 points.

Advice: You may discuss the implementation aspects of this homework with a colleague, the TA or instructor. However, the code must be written individually. *Always* acknowledge sources of information (URL, book, class notes, colleagues, etc.) Please inform instructor quickly about typos or other errors.

---

1. ## Tree graph **(Total 10 points)**

   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).

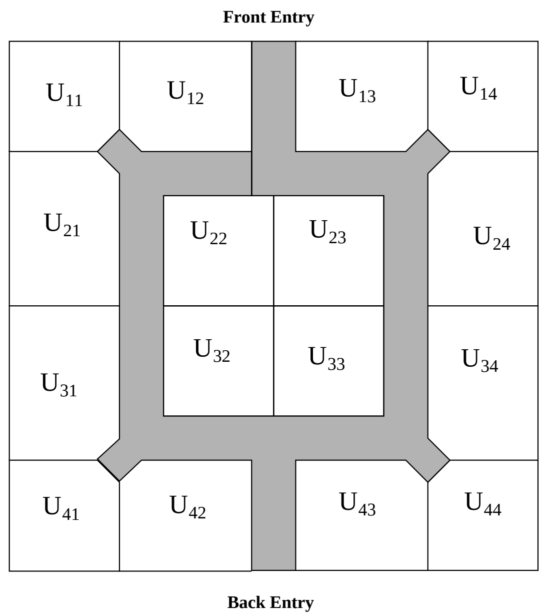
   (a) **(5 points)** What is the width of this graph?

   (b) **(5 points)** State and demonstrate a theorem about the width of a tree-structured graph.

2. ## Vacation house **(Total 10 points)**

   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.

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

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

(c) **(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).

(d) **(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.

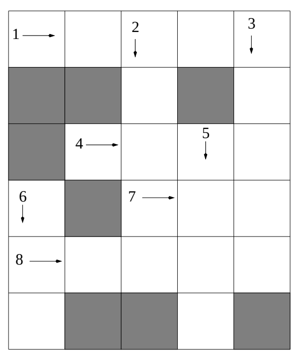
(e) **(Bonus 5 points)** Discuss how unary constraints can be combined with binary constraints to tighten the binary constraints at the preprocessing stage, before search. Give an example of this situation with a unary constraint and a binary constraint in your model, showing, in extension, the effect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

3. Crossword Puzzle **(Total 30 points)**

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.

(a) **(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.

(b) **(5 points)** Draw the constraint graph.

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

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

4. Backtrack search (Total 50 points)

Using the data structures you implemented in Homework 2, implement a chronological backtrack (BT) search procedure and another one with forward-checking (FC) for finding the first solution to the following CSPs:

(a) 6-Queen Problem,

(b) The Race problem,

(c) Crossword puzzle.

Do not run arc-consistency prior to search. Use the static variable ordering heuristic of your choice, preferably one of those you implemented in Homework 2 (i.e., least domain, least degree, and least ratio of domain to degree). Along with your documented code, report the results in a table as the one shown below. **20 points** will be assigned for code clarity and *documentation.*

Your chosen static ordering heuristic						
	BT			FC		
	#CC	#NV	CPU time	#CC	#NV	CPU time
6-Queen **(10 points)**						
Race-problem **(10 points)**						
Crossword puzzle **(10 points)**						

As in Homework 2, you are requested to implement a menu system, which should look like the following:

Enter the number for the program to be run:

1) 6-Queen Problem - Backtrack
2) 6-Queen Problem - Forward checking
3) The Race Problem - Backtrack
4) The Race Problem - Forward checking
5) Crossword puzzle - Backtrack
6) Crossword puzzle - Forward checking

After the number is entered, the selected program should run and the number of constraint checks and nodes visted should be printed to screen, and the menu redisplayed for the next entry. It is usually a good idea to add a menu item to exit the program.

The counter for nodes visted should be incremented every time a variable is instantiated. The counter of constraint checks should be incremented every time two variable-value pairs are checked for consistency.