The University of Melbourne
Department of Computer Science and Software Engineering
Semester 2 Assessment 2005

433-671 Constraint Programming

Time Allowed: 3 Hours
Reading Time: 15 minutes

Authorized materials: Books and calculators are not permitted.

Instructions to Invigilators: One 14 page script. Exam paper may leave the room.

Instructions to students:
This exam counts for 70% of your final grade. There are 6 pages and 5 questions for a total of 70 marks. Attempt to answer all of the questions. Values are indicated for each question and subquestion — be careful to allocate your time according to the value of each question.

Question 1 [11 marks]

The *small sudoko* problem is played on a 4×4 grid rather than a 9×9 grid. Each row and column and sublock of 4 is required to contain exactly the numbers 1 to 4.

1	2	3	4
3	4	1	2
2	1	4	3
4	3	2	1

(a) Define a SICStus Prolog `clpfd` model for the small sudoko problem using the predicate `smallsudoko/1`, which defines the constraints of the small sudoko problem. The argument to this predicate should be a representation of the answer to the small sudoko board. [3 marks].

(b) Discuss the possible complex constraints that could be used in your model. Give an illustration of a partially filled board such that one model could make an inference, where another model could not. [3 marks].

(c) Given the following basic feasible solved form, show the result of a single pivot moving towards the optimal solution:

$$
\begin{array}{lrcrcrcrl}
\text{minimize} & & & 0 & + & 0.5C & - & 0.5E & \text{subject to} \\
& B & = & 3 & - & 0.5C & - & 0.5E & \wedge \\
& D & = & 2 & & & - & E & \wedge \\
& A & = & 4 & & & - & E & \wedge \\
& \multicolumn{8}{l}{A, B, C, D, E \geq 0}
\end{array}
$$

[3 marks].

(d) Explain the role of a simplifier $simpl(C, V)$ in constraint logic programs. Give some desired properties of a simplifier. [2 marks].

Question 2 [13 marks]

(a) Explain how a solver that maintains node, arc and hyper-arc (or domain) consistency will treat the constraint

$$B \neq C \wedge 2A = 3C \wedge A \geq 1 \wedge B \leq 4 \wedge B \geq A + 1$$

given the initial domains are [0..10]. Give domains for A, B, and C after any changes, and explain why the change was made. [4 marks].

(b) Explain how a solver that maintains bounds consistency will treat the constraint

$$B \neq C \wedge 2A = 3C \wedge A \geq 1 \wedge B \leq 4 \wedge B \geq A + 1$$

given the initial domains are [0..10]. Give domains for A, B, and C after any changes, and explain why the change was made. [4 marks].

(c) Give bounds propagation rules for the primitive constraint

$$X \times Y = Z$$

where X,Y and Z take integer values. Give an example domain where the propagation rules for this constraint make no changes, but the resulting domain is not hyper-arc (equivalently domain) consistent. [3 marks].

(d) Discuss the advantages and disadvantages of bounds consistency versus hyper-arc (equivalently domain) consistency. [2 marks].

(e) Singleton consistency works by, for each variable and each possible value in its domain, setting the variable to that value and seeing if the solver detects failure. If it does then the value is clearly impossible and can be removed from the domain of the variable. Give an example of unary and/or binary constraints and variables with domains where singleton consistency will remove a value not removed by arc and node consistency. Using unsafe negation `not (not (...))` give SICStus Prolog `clpfd` code for predicate `sing(V)` to perform singleton consistency on a variable V. Assume that `getDomain(V,L)` returns the current domain of variable V as a list L. [4 marks].

Question 3 [14 marks]

The following SICStus Prolog `clpr` program is meant to define a predicate `mdist(X1,Y1,X2,Y2,D)` where D is the Manhattan distance from $(X1, Y1)$ to $(X2, Y2)$.

```
abs(X,Y) :- ({X >= 0} -> {Y = X} ; {Y = -X}).

mdist(X1,Y1,X2,Y2,D) :-
   {DX = X1 - X2, DY = Y1 - Y2},
   abs(DX,AX),
   abs(DY,AY),
   { D = AX + AY }.
```

(a) Give a non-simplified successful derivation for `abs(-2,E)`. [2 marks].

(b) Give a simplified successful derivation for `mdist(A,0,2,2,D)`. [3 marks.]

(c) The goal `mdist(A,0,-2,2,D), {A =< -3}` finitely fails when it should succeed. Modify the program so that it correctly succeeds. Ensure that all successful derivations for the original version are still successful in the new version. [3 marks].

(d) Give as strong as possible solver redundant constraints that can be placed before the call `abs(A,B)` which may cause failure earlier. [2 marks].

The following code can be used to define predicates for sequences using lists:

```
nonempty([_|_]).

concat([], Y, Y).
concat([A|X], Y, [A|Z]) :- concat(X,Y,Z).
```

The predicate `nonempty(L)` constrains L to be a nonempty sequence. The predicate `concat(L1,L2,L3)` constrains $L3$ to be the concatentation of $L1$ and $L2$.

(e) The goal `nonempty(L1), concat(L1,L2,L), L = []` fails as expected, but `nonempty(L2), concat(L1,L2,L), L = []` does not fail as expected. Explain why? [2 marks].

(f) Using dynamic scheduling we can ensure that both these goals fail. Give a simplified derivation tree for the second goal, using a dynamic literal selection strategy that is finitely failed. [2 marks].

Question 4 [15 marks]

A sheet metal workshop cuts pieces of sheet metal from large sheets of 480 cm x 960 cm. It has received the following order for an amount: it needs to supply 8 pieces of 360 cm x 500 cm, 13 pieces of 240 cm x 360 cm, and 5 pieces of 200 cm x 600 cm.

Amount	dim 1	dim 2
8	360	500
13	240	360
5	200	600

There are only a small number of combinations of cuts from the large sheet that are efficient for the pieces. The different patterns determine how many copies of each pieces can be cut from a single sheet.

Pattern	P1	P2	P3	P4
360 x 500	1	0	0	0
240 x 360	2	2	3	5
200 x 600	0	2	1	0

The different patterns are illustrated in Figure 1.

The aim is to minimize the number of large sheets required to fill the order.

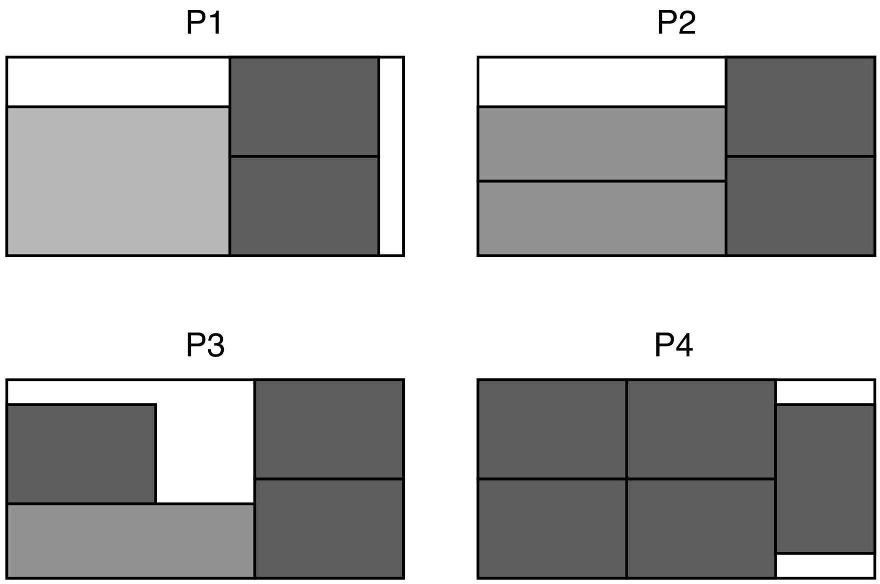

Figure 1: 4 different patterns for cutting the required pieces.

(a) Write a mathematical model of the problem to determine the best cutting strategy to fill the order. Explain the meaning of each problem variable. [3 marks].

(b) Give a SICStus Prolog `clpfd` goal that will find the optimal cutting strategy for the example problem. [2 marks].

(c) Give a data independent version of the mathematical model where the set of patterns and the set of ordered pieces are given as input. [1 mark].

(d) Give a SICStus Prolog `clpfd` program defining the predicate `sheet(Os,Ps,Ns)` which takes as input the list Os of orders of different pieces, the list Ps of pattern descriptions (each a list of the number of each piece in the patter), and returns Ns the numbers of each pattern used to make the optimal order. For example the goal

```
sheet([8,13,5],[[1,2,0],[0,2,2],[0,3,1],[0,5,0]],Ns).
```

asks the example sheet cutting problem above. The goal might return $Ns = [8, 2, 1, 0]$. You can make use of the following predicate definition `delfirstcol` which breaks a list of lists into the list of the first elements, and the list of the rest of the lists.

```
delfirstcol([],[],[]).
delfirstcol([[A]|T], [A|R], []):-
   delfirstcol(T, R, []).
delfirstcol([[A | B] | T], [A | R], [B | T1]):-
   delfirstcol(T, R, T1).
```

For example `delfirstcol([[2,7,6],[9,5,1],[4,3,8]], F, R)` returns `F = [2,9,4]`, `R = [[7,6],[5,1],[3,8]]`. [7 marks].

(e) Which constraint solver: an integer linear programming solver or finite domain propagation solver would be best to tackle your model. Justify your choice. [2 marks].

Question 5 [13 marks]

A simple constraint domain $\mathcal{D}$ over a Booleans, where we use $1 =$ true and $0 =$ false, only allows the following set of primitive constraints: A (A is true), $\neg A$ (A is false) and $C = (A \rightarrow B)$ (C is equivalent to $A \rightarrow B$), where each of A, B and C are Boolean variables.

An example constraint is $A \wedge (C = (A \rightarrow B)) \wedge C$. A valuation in $\mathcal{D}$ is a solution if under the valuation each primitive constraint is satisfied. A solution for the example constraint is $\{A \mapsto 1, B \mapsto 1, C \mapsto 1\}$.

(a) Give a solution of each of the following constraints if one exists, or state it is unsatisfiable.

$$
\begin{array}{ll}
(i) & (C = (A \rightarrow B)) \wedge (C = (B \rightarrow A)) \wedge C \wedge \neg A \\
(ii) & (C = (A \rightarrow B)) \wedge (D = (C \rightarrow E)) \wedge \neg A \wedge (F = (E \rightarrow D)) \wedge \neg F
\end{array}
$$

[2 marks].

(b) Give a constraint in $\mathcal{D}$ which is equivalent to (i) $B = \neg A$ (B is the negation of A) and (ii) $C = (A \wedge B)$ (C is the conjunction of A and B). You need to use additional variables. [2 marks].

(c) A specialised propagation based solver for these constraints keeps track of the variables that are forced to be true and false, and propagates through the complex constraints $C = (A \rightarrow B)$. Define in psuedo-code the propagation function $prop(c, L)$ which takes a single constraint c of the form $C = (A \rightarrow B)$ and a set of literals L defining the variables known to be true and false, and returns the new set of literals known to be true and false. Ensure that it captures all possible (correct) propagations. [3 marks].

(d) Assuming the function *prop* defined in the previous part, give psuedo-code for a correct (incomplete) solver $solv(c)$ which takes a conjunction of primitive constraints in $\mathcal{D}$. [3 marks].

(e) Explain how you could use finite domain variables in SICStus to represent the constraints in $\mathcal{D}$. Give SICStus Prolog `clpfd` goals representing the constraints in a(i) and a(ii). Give the expected answer from each goal. [3 marks].