The University of Melbourne
Department of Computer Science and Software Engineering
Semester 2 Assessment 2006

433-433/633 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 [12 marks]

The *fractions puzzle* is to find 9 different non-zero digits that satisfy

$$\frac{A}{BC} + \frac{D}{EF} + \frac{G}{HI} = 1$$

where BC is shorthand for $10B + C$, EF for $10E + F$ and HI is $10H + I$.

(a) Define a SICStus Prolog `clpfd` model for the fractions puzzle using the predicate `fractions/9`. The arguments to the predicate should be the nine variables in alphabetical order. [3 marks].

(b) There are variable symmetries in the problem: that is if we take a solution we could swap the value of some variables to create another solution. For example swapping the values of $A \leftrightarrow D$, $B \leftrightarrow E$ and $C \leftrightarrow F$ will create a new solution. Give constraints that can be added to the model to remove as many symmetric solutions as possible. [2 marks].

(c) Fouriers algorithm is a projection algorithm for linear inequalities. Show the result of eliminating the variable Y from the conjunction of inequalities:

$$\begin{array}{rcrcrcl} X & + & Y & + & Z & \geq & 0 \\ -X & - & Y & + & Z & \geq & 2 \\ X & & & - & 2Z & \geq & 1 \\ 3X & + & 2Y & & & \geq & 4 \end{array}$$

[3 marks].

(d) Given you wish to project the following conjunction of inequalities onto the variable X, give the best order in which to eliminate variables. *You dont need to do the projection, just give the variable order!* Justify your choice (briefly).

$$\begin{array}{rcrcrcrcrcr} X & + & Y & + & Z & + & T & - & 2U & \geq & 0 \\ -X & - & Y & - & Z & - & T & & & \geq & 2 \\ X & & & - & 2Z & + & 2T & - & U & \geq & 1 \\ 3X & + & 2Y & & & - & 3T & & & \geq & -2 \\ 2X & + & Y & + & Z & & & & & \geq & -2 \\ -X & - & Y & + & Z & - & T & & & \geq & -4 \\ X & & & - & 2Z & + & 2T & - & 3U & \geq & 2 \\ -3X & + & 2Y & & -Z & - & 3T & & & \geq & 7 \\ X & - & Y & + & Z & & & & & \geq & 1 \end{array}$$

[2 marks].

(e) Explain how Fouriers algorithm can be used to construct a simplifier $simpl(C, V)$ for linear inequality constraints. [2 marks].

Question 2 [13 marks]

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

$$Y \leq 5 \land Z \leq 5 \land Z \neq 1 \land X = Y \times Z \land Y \geq Z + 1 \land X \neq 9$$

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

(b) Explain how a solver that maintains bounds consistency will treat the conjunction of primitive constraints

$$Y \leq 5 \land Z \leq 5 \land Z \neq 1 \land X = Y \times Z \land Y \geq Z + 1 \land X \neq 9$$

given the initial domains are [0..10]. Give domains for X, Y, and Z after any changes, and explain why the change was made. Recall the bounds propagation rules for $A = B \times C$ (when they are all non-negative) are

$$\begin{array}{lcl} A & \geq & min_D B \times min_D C \\ A & \leq & max_D B \times max_D C \\ B & \geq & \frac{min_D A}{max_D C} \\ B & \leq & \frac{max_D A}{min_D C} \\ C & \geq & \frac{min_D A}{max_D B} \\ C & \leq & \frac{max_D A}{min_D B} \end{array}$$

[2 marks].

(c) Give bounds propagation rules for the constraint $Y = |X|$, Y is the absolute value of X. [2 marks].

(d) The `element` constraint encodes array lookup. For example, `element`$(I, [3, 0, 2, 1, 3], J)$ means that J takes the Ith value in the list. E.g. when $I = 2$ then $J = 0$. Use reified equations $(B \Leftrightarrow X = Y)$ to model the element constraint `element`$(I, [3, 0, 2, 1, 3], J)$. [2 marks].

(e) The `element` global constraint implements hyper-arc (domain) consistency. Either show an example where your reified version of `element` does not maintain hyper-arc consistency, or (briefly) explain why the reified version does maintain hyper-arc consistency. [3 marks].

Question 3 [14 marks]

The following SICStus Prolog `clpr` program is meant to define a predicate `mg(P,T,I,R,B)` defining the relationship between mortgage parameters: P the principal borrowed, T the number of time periods, I the interest rate paid, R the repayment amount, and B the balance owed at the end.

```
mg(P,T,I,R,B) :- mg(P1,T2,I,R,B),
                 { T >= 1, I > 0, P1 = P * (1 + I) - R, T1 = T - 1}.
mg(P,T,I,R,B) :- { T = 0, P = B }.
```

(a) Give a non-simplified successful derivation for `mg(X,0,0.1,Y,Z)`. [2 marks].

(b) Give a simplified successful derivation for `mg(X,2,Y,Z,0)`. [3 marks.]

(c) The goal `mg(X,2,Y,Z,0)` runs forever using the above program. Modify the program so that its derivation tree is finite. Ensure that all goals with successful derivations of the original program are still have successful derivations of the revised program. [2 marks].

(d) The goal `{X > W, Y <= 2}, mg(X,Y,Z,0,W)` has a successful derivation when it should finitely fail. Modify the program so that it correctly fails. Ensure that all ground calls (where all the arguments are fixed) that have successful derivations for the original version are still successful in the new version. [3 marks].

The following code gives two versions of absolute value constraint $abs(X,Y) \leftrightarrow Y = |X|$.

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

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

(e) (Briefly) Explain the advantages and disadvantages of the two forms versus each other. [2 marks].

(f) Use dynamic scheduling to write the best version of `abs(X,Y)` you can. (Briefly) Explain its strengths. [2 marks].

Question 4 [15 marks]

An amplifier assembly plant has 3 workstations. It has a set of 7 tasks for building an amplifier with durations: [3,6,7,6,4,8,9]. There is a set of precedences on the tasks, one task must be finished before another task can be started. The precedence graph is [(1,2),(1,4),(2,3),(3,5),(3,6),(4,6),(5,7),(6,7)], i.e. task 3 must be finished before task 5 (or 6) can be started. The precedence graph and durations are illustrated in the Figure 1.

The problem is to assign each task to a workstation so that for every arc (a, b) the workstation assigned to task a is less than or equal to the workstation assigned to task b. The aim is to minimize the cycle time which is the maximum over all workstations of the sum of the durations assigned to that workstation.

For example if tasks $\{1, 2, 3, 5\}$, $\{4, 7\}$, $\{6\}$ are assigned to the 3 workstations respectively. The total duration on workstation 1 is $3 + 6 + 7 + 4 = 20$. The assignment violates the

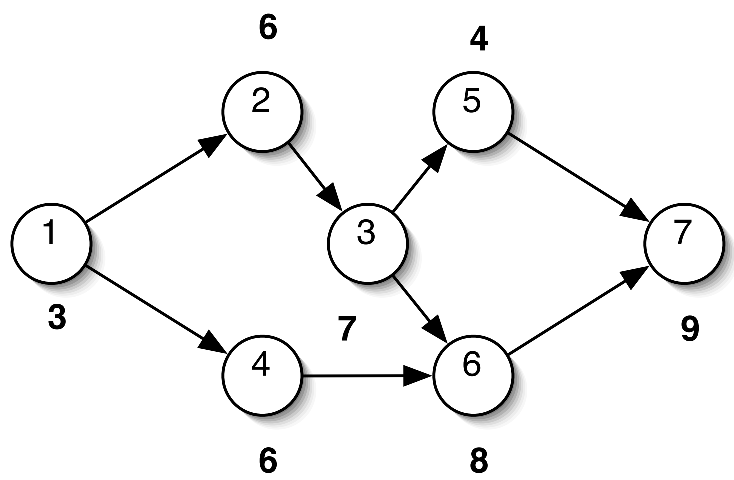

Figure 1: Precedence graph for assembly tasks.

constraints since task 6 is assigned to workstation 3 which is not less than or equal to the workstation (2) assigned to task 7. The remaining constraints are satisfied.

(a) Write a mathematical model of the problem to determine the best task assignment to workstations to minimize cycle time. Explain the meaning of each problem variable. [3 marks].

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

(c) Give a data independent version of the mathematical model where the number of workstations, number of tasks with durations, and precedence graphs are given as inputs. [1 mark].

(d) Give a SICStus Prolog `clpfd` program defining the predicate

```
cycle(NW,Durations,Graph,Cycle)
```

which takes as input the number of workstations NW, the list *Durations* of durations of tasks (the length of the list defines the number), and *Graph* the graph of precedence relations represented as pairs, and returns *Cycle* the minimum cycle time. For example the goal

```
cycle(3,[3,6,7,6,4,8,9],[(1,2),(1,4),(2,3),(3,5),(3,6),(4,6),
(5,7),(6,7)],Cycle).
```

asks the example problem above. The goal might return $Cycle = 13$. [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 [16 marks]

A simple constraint domain $\mathcal{D}$ over a set of totally ordered values S only allows the following set of primitive constraints: $A = B$, $A < B$, $A \leq B$ and $A \neq B$ where each of A and B are variables.

An example constraint is $X < Y \wedge Z \neq X \wedge Z < X$. A valuation in $\mathcal{D}$ is a solution if under the valuation each primitive constraint is satisfied. A solution for the example constraint assuming $S = \{a < b < c < d\}$ is $\{X \mapsto b, Y \mapsto c, Z \mapsto a\}$.

(a) Define a *well-behaved* solver $solv(C)$. [2 marks].

(b) Give a solution of each of the following constraints if one exists, or state it is unsatisfiable, assuming $S = \{a < b < c < d\}$

$$\begin{array}{ll} (i) & Y < T \wedge Z \neq X \wedge X \leq Y \wedge U < Y \wedge Y = Z \wedge X \neq T \\ (ii) & U = V \wedge X = Y \wedge X \neq Z \wedge U < T \wedge Y < U \wedge T < W \wedge Z < U \end{array}$$

[2 marks].

(c) Define a constraint solving algorithm $solv$ for $\mathcal{D}$ assuming the set S is the infinite set of values with no lower or upper bound (that is there is no value $s \in S$ such that $\forall s' \in S. s \leq s'$ or $\forall s' \in S. s' \leq s$). Make the constraint solving algorithm as strong as possible (that it detects unsatisfiability as much as possible). [5 marks]

(d) Comment on how you would change the algorithm if S were instead a finite set. [1 mark].

(e) Define a simplifier $simpl(C, V)$ for $\mathcal{D}$ in the case where S is infinite as in part (b) above. Remember that $simpl(C, V)$ should return a constraint equivalent to C on variables V. [4 marks].

(f) If S is a finite set of integers we can use a propagation based constraint solver to solve $\mathcal{D}$ constraints. Give an example constraint which is unsatisfiable but a propagation based solver would not detect it assuming $S = \{0 < 1 < 2 < \cdots < 100\}$. [2 marks].