First Exam for Course 1DL023:

Constraint Technology for Solving Combinatorial Problems

Autumn 2009, Uppsala University, Sweden

Prepared by Pierre Flener

Wednesday 21 October 2009, from 08:00 to 13:00, in Polacksbacken

Materials: This is a **closed**-book exam. The usage of electronic devices is **not** allowed.

Grading: The grade scale is as follows, when your exam mark (plus any bonus points from the assignments) is x out of 100 exam points:

Swedish Grade	ECTS Grade	Condition
5	A	$90 \leq x \leq 100$
5	B	$80 \leq x \leq 89$
4	C	$65 \leq x \leq 79$
3	D	$58 \leq x \leq 64$
3	E	$50 \leq x \leq 57$
U	FX	$40 \leq x \leq 49$
U	F	$0 \leq x \leq 39$

Help: Normally, an instructor will attend this exam from 10:00 to 11:00.

Answers: Your answers must be written in English. Provide only the requested information and nothing else. Unreadable, unintelligible, and irrelevant answers will not be considered. Be concise and write each answer immediately behind its question and **attach extra solution pages only for Question 3**: if your answer does not fit into the provided space, then it is unnecessarily long and maybe you should re-read the question. Always show **all** the details of your reasoning, and make explicit **all** your assumptions. This question set is double-sided. Circle below which questions you have actually addressed:

Question	Solution Provided?	Max Points	Your Mark
1	yes / no	18	
2	yes / no	30	
3	yes / no	32	
4	yes / no	20	
	Total:	100	

Identity: Name & personal number: ....

Question 1: Stores and Propagators (18 points)

For each of the functions below on stores, answer (**on this page**) the following questions:

a. State (without proof) whether it is a propagator (contracting and monotonic) or not, and give a counterexample if not. **Careful:** Also consider failed stores!

b. Identify (without proof) the implemented constraint, if it is a propagator when it is applied to non-failed stores only.

c. State (without proof) whether it is idempotent or not, and give a counterexample if not.

d. State (without proof) whether it is subsumed (or: entailed) or not after invocation, and give a counterexample if not, assuming it is applied to non-failed stores only.

Careful: Items c. and d. are meaningful even if (you think that) the function is not a propagator! Let $S \setminus T$ denote the subset of elements of set S that do not belong to set T.

A. $p_1(s) = \{\ x \mapsto \emptyset,\ y \mapsto \emptyset\ \}$

   a. Propagator (yes/no+counterexample):

   b. Implements ...

   c. Idempotent (yes/no+counterexample):

   d. Subsumed (yes/no+counterexample):

B. $p_2(s) = \{\ x \mapsto \text{if } |s(y)| = 1 \text{ then } s(x) \setminus s(y) \text{ else } s(x),\ y \mapsto \text{if } |s(x)| = 1 \text{ then } s(y) \setminus s(x) \text{ else } s(y)\ \}$

   a. Propagator (yes/no+counterexample):

   b. Implements ...

   c. Idempotent (yes/no+counterexample):

   d. Subsumed (yes/no+counterexample):

C. $p_3(s) = \{\ x \mapsto s(x) \cap s(y),\ y \mapsto s(x) \cap s(y)\ \}$

   a. Propagator (yes/no+counterexample):

   b. Implements ...

   c. Idempotent (yes/no+counterexample):

   d. Subsumed (yes/no+counterexample):

Question 2: Consistency, Propagation, and Search (30 points)

Consider the following named constraints over decimal digits:

$$x + y \leq 3 \tag{c}$$
$$y + z \leq 4 \tag{d}$$
$$x < y \tag{e}$$
$$y < z \tag{f}$$

Answer (**on this page and on the next two pages**) the following sub-questions:

A. Using the *propagate* "master" algorithm seen in the course, namely the version with events (also known as propagation conditions) and status messages (but without the set MV of modified decision variables), you will perform the pre-search propagation in this sub-question. The following propagation choices are imposed:

- Use **idempotent** propagators achieving **bounds consistency** for all the constraints.
- Post the constraints in the textual order in which they appear above.
- Handle the decision variables in the textual order in which they appear above.
- Use a **first-in first-out queue** for implementing the collection N of propagators that are not known to be at fixpoint.

In other words, and by identifying each propagator with the constraint it implements:

(a) Give (**here**, and without proof) the smallest set of events that trigger the enqueuing of the propagator of each constraint, so that a strictly stronger store might be obtained or that subsumption might be detected:

$es(\text{c}) = \{ \qquad \}$

$es(\text{d}) = \{ \qquad \}$

$es(\text{e}) = \{ \qquad \}$

$es(\text{f}) = \{ \qquad \}$

(b) Fill in the table **on the next page** for the initialisation and every pre-search iteration of *propagate*, where each status message is 'subsumed', 'at fixpoint', or 'not known to be at fixpoint', and each raised event is of the form $any(\alpha)$, $fix(\alpha)$, $min(\alpha)$, or $max(\alpha)$, where α is a decision variable. (Note that you are **not** asked to provide any propagators.)

B. If the pre-search propagation has not solved the problem, then draw the entire search tree (**on the second-next page**). The propagation choices of sub-question 2.A and the following branching heuristics are imposed:

- Use the **left-to-right** variable ordering heuristic (called INT_VAR_NONE in Gecode).
- Use the **bottom-up** value ordering heuristic (called INT_VAL_MIN in Gecode).

Do not expand nodes where all propagators are subsumed. Continue to **use the table on the next page** when propagating a branching decision or a constraint.

$$x + y \leq 3 \quad (c), \quad y + z \leq 4 \quad (d), \quad x < y \quad (e), \quad y < z \quad (f)$$

Chosen prop.	Resulting store	Status message	Raised events	Dependent propagators	Non-subsumed propagators	FIFO queue of non-fixpoint prop.s
init.	$x \mapsto \{\quad\}, y \mapsto \{\quad\}, z \mapsto \{\quad\}$	(not applicable)	(not applicable)	(not applicable)	$\{\quad\}$	$[\quad]$
	$x \mapsto \{\quad\}, y \mapsto \{\quad\}, z \mapsto \{\quad\}$					
	$x \mapsto \{\quad\}, y \mapsto \{\quad\}, z \mapsto \{\quad\}$					
	$x \mapsto \{\quad\}, y \mapsto \{\quad\}, z \mapsto \{\quad\}$					
	$x \mapsto \{\quad\}, y \mapsto \{\quad\}, z \mapsto \{\quad\}$					
	$x \mapsto \{\quad\}, y \mapsto \{\quad\}, z \mapsto \{\quad\}$					
	$x \mapsto \{\quad\}, y \mapsto \{\quad\}, z \mapsto \{\quad\}$					
	$x \mapsto \{\quad\}, y \mapsto \{\quad\}, z \mapsto \{\quad\}$					
	$x \mapsto \{\quad\}, y \mapsto \{\quad\}, z \mapsto \{\quad\}$					
	$x \mapsto \{\quad\}, y \mapsto \{\quad\}, z \mapsto \{\quad\}$					
	$x \mapsto \{\quad\}, y \mapsto \{\quad\}, z \mapsto \{\quad\}$					
	$x \mapsto \{\quad\}, y \mapsto \{\quad\}, z \mapsto \{\quad\}$					
	$x \mapsto \{\quad\}, y \mapsto \{\quad\}, z \mapsto \{\quad\}$					
	$x \mapsto \{\quad\}, y \mapsto \{\quad\}, z \mapsto \{\quad\}$					
	$x \mapsto \{\quad\}, y \mapsto \{\quad\}, z \mapsto \{\quad\}$					
	$x \mapsto \{\quad\}, y \mapsto \{\quad\}, z \mapsto \{\quad\}$					
	$x \mapsto \{\quad\}, y \mapsto \{\quad\}, z \mapsto \{\quad\}$					
	$x \mapsto \{\quad\}, y \mapsto \{\quad\}, z \mapsto \{\quad\}$					
	$x \mapsto \{\quad\}, y \mapsto \{\quad\}, z \mapsto \{\quad\}$					
	$x \mapsto \{\quad\}, y \mapsto \{\quad\}, z \mapsto \{\quad\}$					
	$x \mapsto \{\quad\}, y \mapsto \{\quad\}, z \mapsto \{\quad\}$					
	$x \mapsto \{\quad\}, y \mapsto \{\quad\}, z \mapsto \{\quad\}$					
	$x \mapsto \{\quad\}, y \mapsto \{\quad\}, z \mapsto \{\quad\}$					
	$x \mapsto \{\quad\}, y \mapsto \{\quad\}, z \mapsto \{\quad\}$					
	$x \mapsto \{\quad\}, y \mapsto \{\quad\}, z \mapsto \{\quad\}$					
	$x \mapsto \{\quad\}, y \mapsto \{\quad\}, z \mapsto \{\quad\}$					
	$x \mapsto \{\quad\}, y \mapsto \{\quad\}, z \mapsto \{\quad\}$					
	$x \mapsto \{\quad\}, y \mapsto \{\quad\}, z \mapsto \{\quad\}$					
	$x \mapsto \{\quad\}, y \mapsto \{\quad\}, z \mapsto \{\quad\}$					
	$x \mapsto \{\quad\}, y \mapsto \{\quad\}, z \mapsto \{\quad\}$					
	$x \mapsto \{\quad\}, y \mapsto \{\quad\}, z \mapsto \{\quad\}$					

Space reserved for the search tree of sub-question 2.B: Draw each node of the search tree as a bubble with the current store indicated **inside**. Continue to **use the table on the previous page** when propagating a branching decision or a constraint.

The exam continues on the next page!

Question 3: Modelling (32 points)

Consider the **Progressive Party (PP)** problem. Given a set G of m guest boats, where guest boat $g \in G$ has crew-size $size[g]$, and given a set H of ℓ host boats, where host boat $h \in H$ has spare capacity $cap[h]$ (indicating the number of people that can be hosted, other than the crew of h itself), construct a schedule where the crews of the guest boats party at the host boats over a given number n of periods such that the following constraints are satisfied:

(party-once-a-period) In each period, the crew of each guest boat parties at some host boat.

(host-at-most-once) The crew of each guest boat parties at a particular host boat in at most one period.

(meet-at-most-once) The crews of any two distinct guest boats meet (on the same host boat) in at most one period.

(capacity) The spare capacity of any host boat is not exceeded in any period.

Answer the following sub-questions (**on separate sheets of paper**):

A. Model the PP problem for **any** instance. Show how some, if any, of the constraints named above are automatically enforced by your choice of decision variables. (Hint: Try to achieve this for at least one named constraint, via a two-dimensional matrix of scalar decision variables.) Relate each of your constraints to one of the named constraints above, or declare it to be a channelling constraint. (Hint: Recall that with reification one can transform disjunction into integer inequality.) (20 points)

B. Identify the variable and value symmetries in your **model**. Show how some, if any, of the symmetries of the **problem** are broken by your model. (4 points)

C. Break as many of the symmetries of your model as reasonable. (4 points)

D. Argue for suitable branching heuristics. (4 points)

First read the modelling instructions on the next page! (They are the same as in the exam of autumn 2008.)

Modelling Instructions for Question 3

Your model should be clear and comprehensible, say such that your classmates can understand and implement it without difficulty. Write it in pseudo-code, as on the modelling lecture slides. **The instance data, as well as the decision variables and their domains must be declared**, possibly in mathematical notation, **and their meanings must be given.**

You may use standard mathematical and logical notations, such as:

- $M[i,j]$ (to designate the element in row i and column j of a matrix M; **you may only use this notation when each index is a constant**, or $\star$ in case you want to extract an entire slice of the matrix);
- $\sum_{i \in S} e(i)$ (to designate the sum over all i in the set/enumeration/range S of the numerical expressions $e(i)$);
- $\forall i \in S : \alpha(i)$ (to express that for all i in S, the formula $\alpha(i)$ is true);
- $\wedge$ (and);
- $\leftrightarrow$ (is equivalent to; use this for reification or two-way channelling);
- $\rightarrow$ (implies; use this for one-way channelling).

Try hard to avoid $\vee$ (or). You may **not** use $\exists i \in S : \alpha(i)$ (to express that there exists an i in S such that $\alpha(i)$ is true), **nor** $\exists! i \in S : \alpha(i)$ (to express that there exists exactly one i in S such that $\alpha(i)$ is true), nor negation.

You may **only** use the following global constraints:

- $\textit{distinct}(\{x_1, \ldots, x_n\})$ (also known as *allDifferent*) enforces that any two decision variables x_i and x_j with distinct indices take distinct values, that is $\forall i \neq j \in \{1, \ldots, n\} : x_i \neq x_j$.
- $\textit{element}(\langle a_1, \ldots, a_n \rangle, x, y)$, where $a_1, \ldots, a_n, x, y$ are integers or decision variables, enforces that y is equal to the x^{th} element of the sequence $\langle a_1, \ldots, a_n \rangle$, that is $a_x = y$.
- $\textit{gcc}(\langle x_1, \ldots, x_n \rangle, \langle v_1, \ldots, v_m \rangle, \langle \textit{min}_1, \ldots, \textit{min}_m \rangle, \langle \textit{max}_1, \ldots, \textit{max}_m \rangle)$ enforces that the number of elements in the decision-variable sequence $\langle x_1, \ldots, x_n \rangle$ that take the constant value v_j is between the integer constants $\textit{min}_j$ and $\textit{max}_j$ inclusive, for all $j \in 1, \ldots, m$.
- $\textit{lex}(\langle x_1, \ldots, x_n \rangle, \langle y_1, \ldots, y_n \rangle)$ enforces that the decision-variable sequence $\langle x_1, \ldots, x_n \rangle$ is lexicographically smaller than or equal to the decision-variable sequence $\langle y_1, \ldots, y_n \rangle$.
- $\textit{linear}(\langle c_1, \ldots, c_n \rangle, \langle x_1, \ldots, x_n \rangle, R, d)$ enforces that the scalar product of the integer sequence $\langle c_1, \ldots, c_n \rangle$ with the decision-variable sequence $\langle x_1, \ldots, x_n \rangle$ is in relation R with the integer d, where $R \in \{<, \leq, =, \neq, \geq, >\}$, that is $\left(\sum_{i=1}^{n} c_i \cdot x_i\right)\ R\ d$.

The exam continues on the next page!

Question 4: Global Constraints (20 points)

Consider the instance below of the Sudoku puzzle, where the task is to fill the cells of the 9×9 grid with integer values in the range $1, \ldots, 9$ such that the cells of each row, each column, and each highlighted 3×3 block are pairwise different:

	6		1		4		5	
		8	3		5	6		
2								1
8			4		7			6
		6				3		
7			9		1			4
5						u	v	2
		7	2		6	9	w	x
	4		5		8	y	7	z

Answer (**on this page**) the following sub-questions, on the straightforward constraint model of this problem (seen in the course):

A. Propagate **each** constraint to **value consistency** (in order to make the next sub-question more interesting than it would otherwise be); only indicate the domains in the resulting store for the decision variables $u, v, \ldots, z$ of the bottom-right 3×3 block: (3 points)

$\{u \mapsto \{\quad\},\ v \mapsto \{\quad\},\ w \mapsto \{\quad\},\ x \mapsto \{\quad\},\ y \mapsto \{\quad\},\ z \mapsto \{\quad\}\}$

B. Continuing from your store of sub-question A, propagate **the** constraint on the bottom-right 3×3 block to **domain consistency** using Régin's propagator: (12 points)

Chosen matching: $u = \quad ,\ v = \quad ,\ w = \quad ,\ x = \quad ,\ y = \quad ,\ z =$

Alternating paths:

Alternating cycles:

Vital edges:

Pruned edges:

$\{u \mapsto \{\quad\},\ v \mapsto \{\quad\},\ w \mapsto \{\quad\},\ x \mapsto \{\quad\},\ y \mapsto \{\quad\},\ z \mapsto \{\quad\}\}$

C. What does Régin's propagator do if there are more decision variables than values? (2 points)

D. Does Régin's propagator always compute a fixpoint? Why? (1 point)

E. When can Régin's propagator signal entailment? Why? (2 points)