Exam in Constraint Programming (1DL440)

Prepared by Pierre Flener

Thursday 16 December 2010, from 08:00 to 13:00, in Gimogatan 4, sal 1

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 or project) is $x$ out of 100 exam points:

<table>
<thead>
<tr>
<th>Swedish Grade</th>
<th>ECTS Grade</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>A</td>
<td>$90 \leq x \leq 100$</td>
</tr>
<tr>
<td>5</td>
<td>B</td>
<td>$80 \leq x \leq 89$</td>
</tr>
<tr>
<td>4</td>
<td>C</td>
<td>$65 \leq x \leq 79$</td>
</tr>
<tr>
<td>3</td>
<td>D</td>
<td>$58 \leq x \leq 64$</td>
</tr>
<tr>
<td>3</td>
<td>E</td>
<td>$50 \leq x \leq 57$</td>
</tr>
<tr>
<td>U</td>
<td>FX</td>
<td>$40 \leq x \leq 49$</td>
</tr>
<tr>
<td>U</td>
<td>F</td>
<td>$0 \leq x \leq 39$</td>
</tr>
</tbody>
</table>

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 4: if an 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:

<table>
<thead>
<tr>
<th>Question</th>
<th>Solution Provided?</th>
<th>Max Points</th>
<th>Your Mark</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>yes / no</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>yes / no</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>yes / no</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>yes / no</td>
<td>38</td>
<td></td>
</tr>
<tr>
<td>Total:</td>
<td></td>
<td>100</td>
<td></td>
</tr>
</tbody>
</table>

Identity: Your anonymous exam code: .........................................................
Question 1: Constraint Programming Technology (12 points)

If you met the head teacher again in a few years and wanted to tell him what you have remembered from this course, how would you succinctly formulate (only within the allocated space on this page):

A. The essential features of constraint-based complete search. (5 points)

B. The essential features of constraint-based local search. (5 points)

C. The relationship between these essential features of these two approaches to combinatorial problem solving offered by constraint programming technology. (2 points)
Question 2: Consistency, Propagation, and Search  (30 points)

Consider the following named constraints over the digits of base 10:

\[ x + z \leq 4 \]  
\[ \text{element}(2, 1, 6, 2, 1, 0), x, y \]  
\[ y \leq 5 \]  
\[ \text{distinct}(\{x, y, z\}) \]

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

A. Using the *propagate* fixpoint 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), perform the pre-search propagation to compute the root of the search tree. The following propagation choices are imposed:

- Use *idempotent* propagators achieving *bounds(Z) consistency* on the arithmetic constraints and *domain consistency* on the other constraints.
- Post the constraints in the textual order in which they appear above.
- Handle the decision variables in the alphabetical order on their names.
- Use a *first-in first-out queue* (FIFO) for implementing the set \( N \) of propagators that are not known to be at fixpoint. Note that \( N \) is not a multi-set.

In other words, and by denoting the propagator of constraint \( \gamma \) also by \( \gamma \), do the following:

(a) Give (only in the four lines after this paragraph, and without proof) the *smallest* set \( es(\gamma) \) of events that trigger the enqueuing of the propagator \( \gamma \) of each constraint \( \gamma \), so that a strictly stronger store might be obtained when the propagator \( \gamma \) is woken up:

\[
\begin{align*}
es(c) &= \{ \} \\
es(d) &= \{ \} \\
es(e) &= \{ \} \\
es(f) &= \{ \}
\end{align*}
\]

(b) Fill in the table on the next page for the initialisation and every pre-search iteration of *propagate*, where each status message is either ‘subsumed’, or ‘at fixpoint’, or ‘unknown’ (short for ‘not known to be at fixpoint’), or ‘failed’ (when the domain of some decision variable becomes empty), and each raised event is of the form *any*(\( \alpha \)), *fix*(\( \alpha \)), *min*(\( \alpha \)), or *max*(\( \alpha \)), where \( \alpha \) is a decision variable. Write ‘(none)’, \{\} (or \( \emptyset \)), or \[ \], rather than nothing, where appropriate, and keep in mind that \( \{\emptyset\} \neq \emptyset \). As specified on the last page of this exam, assume here that the array argument of the *element* constraint is *indexed from 1*, not from 0. Note that you are not asked to provide any propagators.
\( x + z \leq 4 \) (c), \( \text{element}(2, 1, 6, 2, 1, 0), x, y \) (d), \( y \leq 5 \) (e), \( \text{distinct}(x, y, z) \) (f)

<table>
<thead>
<tr>
<th>Chosen prop.</th>
<th>Resulting store</th>
<th>Status message</th>
<th>Smallest set of raised events</th>
<th>Dependent propagators ( DP )</th>
<th>Non-subsumed propagators ( P )</th>
<th>FIFO queue ( N ) of non-fixpoint prop.s</th>
</tr>
</thead>
<tbody>
<tr>
<td>(none)</td>
<td>( x \mapsto { }, y \mapsto { }, z \mapsto { } )</td>
<td>(not applicable)</td>
<td>(not applicable)</td>
<td>(not applicable)</td>
<td>{ }</td>
<td>{ }</td>
</tr>
<tr>
<td>1</td>
<td>( x \mapsto { }, y \mapsto { }, z \mapsto { } )</td>
<td>(not applicable)</td>
<td>{ }</td>
<td>{ }</td>
<td>{}</td>
<td>{}</td>
</tr>
<tr>
<td>2</td>
<td>( x \mapsto { }, y \mapsto { }, z \mapsto { } )</td>
<td>(not applicable)</td>
<td>{ }</td>
<td>{ }</td>
<td>{}</td>
<td>{}</td>
</tr>
<tr>
<td>3</td>
<td>( x \mapsto { }, y \mapsto { }, z \mapsto { } )</td>
<td>(not applicable)</td>
<td>{ }</td>
<td>{ }</td>
<td>{}</td>
<td>{}</td>
</tr>
<tr>
<td>4</td>
<td>( x \mapsto { }, y \mapsto { }, z \mapsto { } )</td>
<td>(not applicable)</td>
<td>{ }</td>
<td>{ }</td>
<td>{}</td>
<td>{}</td>
</tr>
<tr>
<td>5</td>
<td>( x \mapsto { }, y \mapsto { }, z \mapsto { } )</td>
<td>(not applicable)</td>
<td>{ }</td>
<td>{ }</td>
<td>{}</td>
<td>{}</td>
</tr>
<tr>
<td>6</td>
<td>( x \mapsto { }, y \mapsto { }, z \mapsto { } )</td>
<td>(not applicable)</td>
<td>{ }</td>
<td>{ }</td>
<td>{}</td>
<td>{}</td>
</tr>
<tr>
<td>7</td>
<td>( x \mapsto { }, y \mapsto { }, z \mapsto { } )</td>
<td>(not applicable)</td>
<td>{ }</td>
<td>{ }</td>
<td>{}</td>
<td>{}</td>
</tr>
</tbody>
</table>
B. If the pre-search propagation of sub-question 2.A has not solved the problem, then draw the search tree (only on this page), with constraint stores as nodes. 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 previous page when propagating a branching decision or a constraint, and mark there the starting row of each call to propagate.
Question 3: 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 set \{1, \ldots, 9\} such that the values of each row, each column, and each highlighted 3 × 3 block are pairwise different:

3 5 4
8 9 4 2 3
4 2

\[
\begin{array}{ccc}
8 & v & 4 & w & 1 & 5 \\
9 & 7 & x & 3 & 4 \\
7 & 1 & y & 2 & z & 9 \\
8 & 1 & 4 & 9 \\
6 & 4 & 1 & 7 & 8 \\
9 & 2 \\
\end{array}
\]

Answer (only on this page) the following sub-questions, on the straightforward constraint model of this puzzle (seen in the course, namely the one with 3 · 9 distinct constraints):

A. Propagate each constraint once to value consistency (in order to make the next sub-question more interesting than it would be if we propagated them to their greatest common fixpoint under domain consistency), by processing first the row constraints in top-down order, then the column constraints in left-to-right order, and finally the box constraints in left-to-right top-down order; only indicate the domains in the resulting start store for the decision variables \(v, w, x, y, z\) of the central 3 × 3 block:

\[
\text{Start store: } \{v \mapsto \{\}, \ w \mapsto \{\}, \ x \mapsto \{\}, \ y \mapsto \{\}, \ z \mapsto \{\}\}
\]

B. Continuing from your start store of sub-question 3.A, propagate to domain consistency using Régis’s propagator the unique distinct constraint that operates exactly on the central 3 × 3 block:

\[
\text{Chosen maximal matching: } v = , \ w = , \ x = , \ y = , \ z = \\
\text{Alternating paths:} \\
\text{Alternating cycles:} \\
\text{Vital edges:} \\
\text{Pruned edges:} \\
\text{New store: } \{v \mapsto \{\}, \ w \mapsto \{\}, \ x \mapsto \{\}, \ y \mapsto \{\}, \ z \mapsto \{\}\}
\]

C. Can there be free nodes in the variable-value graphs for a Sudoku puzzle? Why? (2 points)

D. Can there be alternating paths in the variable-value graphs for a Sudoku puzzle? (1 point)

E. How is Régis’s propagator incremental when another propagator removes for some decision variable another value than the one it had in the chosen maximal matching? (2 points)
Question 4: Modelling (38 points)

Consider the Progressive Party (PP) problem. Given a set $G$ of $m$ guest boats, where guest boat $g \in G$ has crew size $\text{size}[g]$, and given a set $H$ of $\ell$ host boats, where host boat $h \in H$ has spare capacity $\text{cap}[h]$ (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 together at some host boat.
- **(host-at-most-once)** The crew of each guest boat parties together 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 (not here, but on attached 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.)

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.

C. Break as many of the symmetries of your model as reasonable.

D. Argue for suitable branching heuristics.

*First read the modelling instructions on the next page!*
(They are essentially the same as in the exam of autumn 2009, except that the reifying red parentheses are now outlawed.)

*We will not grade anything written on this or the next page!*
Modelling Instructions for Question 4

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 and their semantics must be given, and every constraint must be annotated with an English paraphrase.

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

- $M[i,j]$, to designate the element in row $i$ and column $j$ of a matrix $M$; unlike in the homeworks, you may here only use this notation when each index is a constant, or $*$ in case you want to extract an entire slice of the matrix;

- $\sum_{i \in S} f(i)$, to designate the sum over all $i$ in set $S$ of the numerical expressions $f(i)$;

- $\forall i \in S : \alpha(i)$, to express that for all $i$ in set $S$, the formula $\alpha(i)$ is true;

- $\&$ (logical and);

- $\Leftrightarrow$ (is logically equivalent to); you may only use this for reification or two-way channelling;

- $\Rightarrow$ (logically implies); you may only use this for one-way channelling.

Try hard to avoid $\lor$ (logical or). You may use neither the implicitly reifying (so-called “red”) parentheses for formulating higher-order constraints, nor $\exists i \in S : \alpha(i)$ to express that there exists at least one $i$ in set $S$ such that formula $\alpha(i)$ is true, nor $\exists! i \in S : \alpha(i)$ to express that there exists exactly one $i$ in set $S$ such that formula $\alpha(i)$ is true, nor $\neg$ (logical negation).

You may only use the following global constraints:

- $\text{distinct}(\{x_1,\ldots,x_n\})$, also known as $\text{allDifferent}$, enforces that any two decision variables $x_i$ and $x_j$ with distinct indices take distinct values, that is $\forall i \neq j \in \{1,\ldots,n\} : x_i \neq x_j$.

- $\text{element}(\langle a_1,\ldots,a_n \rangle, x, y)$, where $a_1,\ldots,a_n, x, y$ are integers or decision variables, enforces that $y$ is equal to the element at position $x$ of the array $\langle a_1,\ldots,a_n \rangle$, that is $a_x = y$.

- $\text{gcc}(\{x_1,\ldots,x_n\}, \langle v_1,\ldots,v_m \rangle, \langle \text{min}_1,\ldots,\text{min}_m \rangle, \langle \text{max}_1,\ldots,\text{max}_m \rangle)$ enforces that the number of decision variables among $\{x_1,\ldots,x_n\}$ that take the constant value $v_j$ is between the integers $\text{min}_j$ and $\text{max}_j$ inclusive, for all $j \in \{1,\ldots,m\}$.

- $\text{lex}(\langle x_1,\ldots,x_n \rangle, \langle y_1,\ldots,y_n \rangle)$ enforces that the decision-variable array $\langle x_1,\ldots,x_n \rangle$ is lexicographically smaller than or equal to the decision-variable array $\langle y_1,\ldots,y_n \rangle$.

- $\text{linear}(\langle c_1,\ldots,c_n \rangle, \langle x_1,\ldots,x_n \rangle, R, d)$ enforces that the scalar product of the integer array $\langle c_1,\ldots,c_n \rangle$ with the decision-variable array $\langle x_1,\ldots,x_n \rangle$ is in relation $R$ with the integer $d$, where $R \in \{<,\leq,=,\neq,\geq,>\}$, that is $\left(\sum_{i=1}^n c_i \cdot x_i\right) R d$.

Good Luck!