Combinatorial Optimisation using Constraint Programming (course 1DL441)  
Uppsala University – Autumn 2015  
Assignment 2: Consistency, Propagation, and Search

Prepared by Pierre Flener  
— Deadline: 18:00 on Thursday 8 October 2015 —

The objective of this assignment is to understand in depth how a constraint solver works during the initial propagation and during search.

Consider the following constraint satisfaction problems (CSPs):

\[
\begin{align*}
  c_1: & x < y \\
  c_2: & x + y < 8 \\
  c_3: & y > z \\
  & x \mapsto \{2, \ldots, 10\} \\
  & y \mapsto \{0, \ldots, 10\} \\
  & z \mapsto \{3, \ldots, 10\} \\
  c_4: & \text{ELEMENT}([2, 1, 8, 2, 1, 0], x, y) \\
  c_5: & x + y \leq 7 \\
  c_6: & z \cdot z \leq 5 \\
  c_7: & \text{DISTINCT}([x, y, z]) \\
  & x, y, z \mapsto \{1, \ldots, 9\}
\end{align*}
\]

For each of these CSPs, perform the following sequence of tasks:

A. Using the Propagate fixpoint algorithm seen in the course, namely the version with propagator conditions and status messages (but without the set \texttt{ModVars} of decision variables whose domains have been modified), perform the pre-search propagation to compute the root of the search tree. The following choices are imposed:

- Use \textit{idempotent} propagators achieving \textit{bounds(D)} consistency for the arithmetic constraints and \textit{domain} consistency for the other 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 in the stores.
- Use a \textit{first-in first-out queue} (FIFO) for implementing the set \textit{Q} of propagators that are not known to be at fixpoint. (Note that \textit{Q} is \textit{not} a multi-set.)

In other words, upon denoting the propagator of constraint \( c_i \) by \( p_i \), perform the following sequence of sub-tasks, noting that you are \textbf{not} asked to design any propagators:

(a) For each constraint \( c_i \), give (without proof) the set \text{PropConds}(p_i) of conditions that should trigger the scheduling of the propagator \( p_i \), as a strictly stronger store might then be obtained. Each propagator condition is of the form ‘Any(\( \alpha \))’, ‘Fixed(\( \alpha \))’, or ‘Bounded(\( \alpha \))’, where \( \alpha \) is a decision variable of \( c_i \). Write ‘(none)’ rather than nothing, where appropriate.

\footnote{For the right-hand CSP, for teams of one person.}
(b) Fill in a table like the one on the next page for the initialisation (in the first row) and every pre-search iteration of Propagate. Each status message is as precise as possible, the options being ‘Subsumed’, ‘AtFixpt’, ‘Unknown’ (short for “not known to be at fixpoint”), and ‘Failed’). Each modification event is of the form ‘None(α)’, ‘Failed(α)’, ‘Fixed(α)’, ‘Bounded(α)’, or ‘Any(α)’, where α is a decision variable of the propagated constraint. Write ‘(none)’ rather than nothing, where appropriate. The array argument of the Element constraint is indexed from 1, not from 0.

(c) Does anything change in your answers to the previous sub-tasks when instead achieving domain consistency for all the constraints? Why? (Note that you are not asked to fill in another table.)

B. If the pre-search propagation has not solved the CSP, then draw (possibly by hand, but neatly) the search tree with all the solutions, with pairs of non-subsumed propagator sets and constraint stores as nodes, and decisions (which are constraints) as labelled arcs. The choices of Task A and the following branching heuristics are imposed:

- Use largest-minimum variable selection (called INT_VAR_MIN_MAX in Gecode).
- Use bottom-up value selection (called INT_VAL_MIN in Gecode).

Continue to use a table like the one of Task A when propagating a branching decision or a constraint, and mark there the starting row of each call to Propagate.

Submission Instructions

All question answers must be in a single report in PDF format; all other formats are rejected.

- Take the demo report for Assignment 2 at http://www.it.uu.se/edu/course/homepage/consprog/ht15/resources/demoReport-A2 as a strict guideline for the structure.
- Write clear answers, state any assumptions you make, and justify all claims and answers.
- Thoroughly proof-read, spell-check, and grammar-check your report.

Only one of the teammates submits the report, without folder structure and without compression, via the Student Portal (whose clock may differ from yours) by the given hard deadline. For team t, the report must be called assignment2-teamt.pdf.

Grading

If all tasks have been seriously attempted, then you get an initial score of at least 1 point:

- If your answers are mostly correct, then you get an initial score of 4 or 5 points; you are not invited to the grading session and your initial score is your final score.

- If your answers have many errors, then you get an initial score of 1 or 2 points; you are invited to the grading session, where you can increase your initial score by 1 point into your final score.

Otherwise you get a final score of 0 points and fail the Assignment part of the course.
<table>
<thead>
<tr>
<th>Chosen prop.</th>
<th>Resulting store</th>
<th>Status message</th>
<th>Modification events</th>
<th>Dependent prop.s $DepProps$</th>
<th>Non-subsumed propagators $R$</th>
<th>FIFO queue $Q$ of propagators</th>
</tr>
</thead>
<tbody>
<tr>
<td>(none)</td>
<td>$x \mapsto {}$, $y \mapsto {}$, $z \mapsto {}$</td>
<td>(none)</td>
<td>(none)</td>
<td>(none)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x \mapsto {}$, $y \mapsto {}$, $z \mapsto {}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x \mapsto {}$, $y \mapsto {}$, $z \mapsto {}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x \mapsto {}$, $y \mapsto {}$, $z \mapsto {}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x \mapsto {}$, $y \mapsto {}$, $z \mapsto {}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x \mapsto {}$, $y \mapsto {}$, $z \mapsto {}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x \mapsto {}$, $y \mapsto {}$, $z \mapsto {}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x \mapsto {}$, $y \mapsto {}$, $z \mapsto {}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x \mapsto {}$, $y \mapsto {}$, $z \mapsto {}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x \mapsto {}$, $y \mapsto {}$, $z \mapsto {}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x \mapsto {}$, $y \mapsto {}$, $z \mapsto {}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x \mapsto {}$, $y \mapsto {}$, $z \mapsto {}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>