Foundations of Constraint Programming

Prof. Michael Thielscher, Sebastian Voigt

International Master Program in Computational Logic — winter term 2008/2009

Date of exercise: 10.12.2008

Exercise 4.1

Consider the following CSP:

$$\langle x \neq 10, x = y + 1, all\_different(x, y, z), -x - y - z = w;$$
$$x \in [10 \ldots 13], y \in [10 \ldots 12], z \in [10 \ldots 12], w \in [-32 \ldots - 30]\rangle$$

a) Is this CSP consistent? Is is node consistent, arc consistent, directionally arc consistent, path consistent, directionally path consistent? Is it hyper-arc consistent?

b) Can you find some k for which this CSP is not k-consistent ? If yes, explain why.

c) Are there instances of i, m such that the CSP is not relationally (i, m)-consistent ? If yes, show why and what are the differences to the notion of k-consistency.

Exercise 4.2

a) Prove the Note on Slide IV/15:

A normalized CSP is path consistent iff for each subsequence x, y, z of its variables

$$C_{x,y} \subseteq C_{x,z} \cdot C_{y,z}^T, \qquad C_{x,z} \subseteq C_{x,y} \cdot C_{y,z}, \qquad C_{y,z} \subseteq C_{x,y}^T \cdot C_{x,z}.$$

b) Prove the first note on Slide IV/25:

A node consistent CSP is arc consistent iff it is 2-consistent.

Hint: You need to assume that the CSP does not contain the false constraint $\perp$.

Exercise 4.3

a) Implement an Eclipse-Prolog-predicate `permutation(A,B)` that generates a permutation B of a list A. All permutations can be computed by backtracking using ";" when prompted. Use the following queries to check that your program works correctly:

```
:-permutation([1,2,3],B).
:-permutation(A,B).
```

b) Use the constraint solving library `ic` to write a predicate `sorted(L)` for a list L that is using constraints and is true if the elements of the list are sorted in ascending order.

Implement a predicate `permsort(L,SL)` that takes a list L and generates a sorted list SL by testing permutations until a sorted permutation is found.

c) Redefine `permsort` such that it first calls `sorted` and then `permutation`. Compare the two ways of implementing `permsort` with respect to run time.

Check that `permsort` also works for lists with variables, for example `permsort([1,A,3],SL)`.