Universität Ulm
Fakultät für Informatik
Prof. Dr. Thom Frühwirth

Constraint-Programmierung, WS 02/03
Übungsblatt 4

Abgabe bis Mittwoch, 27.11., 10:00 Uhr

Nebenläufige CLP

Wir verwenden eine Untermenge von Constraint Handling Rules (CHR), um in NCLP zu programmieren. Eine NCL-Klausel der Form $H \leftarrow C : D \mid G$ wird in CHR geschrieben als `N @ H <=> C & D | G`, wobei `N @` ein optionaler Name für die Regel ist.

Lesen Sie die Kurzbeschreibung zu CHR (siehe Constraint-Buch-Webseite `www.pst.informatik.uni-muenchen.de/~fruehwir/buch.html`*).*

Vergleichen Sie das Verhalten der folgenden NCLP- und CLP-Programme, die aus jeweils einer der gegebenen Regeln bestehen, indem Sie jeweils die unten angegebenen Ziele auswerten. Überprüfen Sie Ihre Antworten am Rechner und machen Sie sich klar, warum man unterschiedliche Ergebnisse erhält.

```
p1 @ p(a) <=> true | true.
p2 @ p(X) <=> X=a | true.
p3 @ p(X) <=> true | X=a.
p4 @ p(X) <=> true & X = a | true.
p5 @ p(X) <=> X = a & X = b | true.
p6 @ p(a) :- true.
p7 @ p(X) :- X=a.
p8 @ p(X) :- X = a, X = b.
```

Ziele (Anfragen): `p(a)`, `p(b)` und `p(C)`, jeweils einzeln.

Aufgabe 4-1

Implementieren Sie die drei Varianten des NCLP-Beispiels `min` (minimum) aus der Vorlesung (siehe NCLP-Folien, via Constraint-Buch-Webseite erreichbar) in CHR. Machen Sie sich das unterschiedliche Verhalten der Varianten anhand von folgenden sechs Anfragen klar:

```
min(1,2,C).
min(A,2,1).
min(A,2,3).
min(A,A,B).
min(1,2,1).
min(1,2,3).
```

Aufgabe 4-2 Freiwillig

Implementieren Sie das NCLP-Beispiel `hamming` aus der Vorlesung (siehe NCLP-Folien, via Constraint-Buch-Webseite erreichbar) in CHR. Um die Ergebnisse der unendlichen Zahlenfolge sichtbar zu machen, sobald sie vorliegen, schreiben Sie ein zusätzliches rekursives Prädikat `observe`, das das Ergebnis von `hamming` beobachtet und ausgibt (`write(...),nl`). Der Aufruf sieht dann so aus: `observe(S),hamming(S)`.