Exercise 3

Foundations of Constraint Programming

Prof. Michael Thielscher, Sebastian Voigt

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

Date of exercise: 26.11.2008

Exercise 3.1

Given a substitution $\theta := \{x_1/t_1, \ldots, x_n/t_n\}$, let the set of variables $\{x_1, \ldots, x_n\}$ be denoted by $Dom(\theta)$, the set of terms $\{t_1, \ldots, t_n\}$ by $Range(\theta)$ and the set of variables that occur in a term from $Range(\theta)$ by $Ran(\theta)$. Moreover a substitution θ is called **idempotent** if $\theta = \theta\theta$.

Prove that θ is idempotent iff $Dom(\theta) \cap Ran(\theta) = \emptyset$. Conlude that if a derivation using the Unif rules is successful, then the final CSP determines an idempotent substitution.

Exercise 3.2

Take the following set of linear equations:

$$\begin{array}{rcrcrcl} a & + & b & + & c & = & 0 \\ 4a & + & 2b & + & c & = & 1 \\ 9a & + & 3b & + & c & = & 3 \end{array}$$

Apply the rules of the Lin proof system (see Slides 23/24 in Chapter 3) to compute an mgu for this set of equations.

Exercise 3.3

A magic square of size n is an $n \times n$ matrix containing the numbers from 1 to n^2 (each number exactly once) such that the sums of each row, each column and the two main diagonals are equal. Write a program in Eclipse-Prolog which generates magic squares of size n using the constraint solving library `ic` (load it with `:- lib(ic).`). Solve this task via the following subtasks:

a) Use a list of lists data structure. For example, $[[2,7,6],[9,5,1],[4,3,8]]$ represents a magic square of size $n = 3$ where the first row is 276, the first column is 294 and the upper left to lower right diagonal is 258.

   Define a predicate `assignM(Matrix,N,Nsq)` which instantiates a fresh variable $Matrix$ to a list of N rows, where each row itself is a list containing N variables with domain $[1..Nsq]$.

b) Define a predicate `rowC(Matrix,Sum)` (`colC(Matrix,Sum)`, `diagC(Matrix,Sum)`, resp.) which adds constraints such that the sum of each row (column, diagonal, resp.) equals Sum.

   Hint: Use the predefined predicate `sumlist/2` from the library `ic_global` which can be loaded with `:- import sumlist/2 from ic_global.`

c) Define a predicate `square(N)` which shows one solution of a magic square of size N and define a predicate `allsquare(N)` which shows the number of all existent solutions of this size.