Technische Universität München *Summer 2011*
Fakultät für Informatik *Kalmer Apinis*
Prof. Dr. H. Seidl *apinis@in.tum.de*

Virtual Machines

Exercise Sheet 10

Deadline: July 22th, 2011

Exercise 1: *Most General Unifiers* *10 Points*

When two terms ought to be unified, first they have to be compared. If both are atoms, then the unification succeeds if and only if they are identical. In case both terms are variables, one is bound to the other and unification succeeds. Should one be some variable X and the other be some structure t, then unification succeeds while binding X to t if and only if X does not occur in t. Lastly, if both terms are structures, then each pair of sub-terms has to be unified recursively and the whole unification succeeds if and only if each pair of sub-terms could be properly unified.

The result of unification is either success or failure together with a set of variable bindings, known as a *unifier*. There may be many such unifiers for any pair of terms, yet there will be at most one single *most general unifier*. Other unifiers simply add extra bindings for sub-terms which are variables in the original terms.

Determine the most general unifiers for the following pairs of terms wherever possible or explain why unification fails:

a) $f(X, g(Y, b))$ and $f(g(a, Z), X)$
b) $f(g(a, Z), X)$ and $f(X, g(b, Y))$
c) $g(X, f(X, X))$ and $g(f(a, a), f(Y, Y))$
d) $g(g(X, g(a, Z)), g(f(V), V))$ and $g(g(f(Y), Y), g(Z, g(Y, Z)))$
e) $a(b, X, d(e, Y, g(h, i, Z)))$ and $a(U, c, d(V, f, g(W, i, j)))$
f) $f(X, 5, Y, x(a, g(6, 7)))$ and $f(Y, 5, c, x(Z, g(6, X)))$

Example: The most general unifier of $f(X, g(Y, b), Z)$ and $f(g(a, U), g(a, V), W)$ is $[X/g(a, U), Y/a, Z/W, V/b]$. The terms $f(X, Y)$ and $f(g(Y), g(X))$ have no unifiers.

Exercise 2: *Reference Chains* *3 Points*

According to the lecture, a runtime function `deref` was defined which contracts reference chains by maximal dereference.

a) When do non-trivial reference chains appear, i.e. such that `deref` is called recursively at least once? Give an example!
b) Explain how long such reference chains can become.

Exercise 3: *WiM Code Generation* *7 Points*

Consider the following Prolog program:

```
reverse(L1, R, L2) :- L1 = [], L2 = R.
reverse(L1, R, A) :- L1 = [X|L], reverse(L, [X|R], A).
reverse(L, R) :- reverse(L, [], R).
?- reverse(X, [4,2,1]).
```

a) Translate the program to WiM code (without optimization).

b) Execute the WiM code, showing the sequence of (sub-)goals being called in conjunction with the stack and the heap after each of these goals have been processed. Where is backtracking done?