**Übung Künstliche Intelligenz SS 2008**
**Wirtschaftsinformatik und Maschinelles Lernen (ISMLL)**
**Prof. Dr. Dr. Lars Schmidt-Thieme, Krisztian Buza**

# Übungsblatt 9

Abgabe: Freitag, 27.06.2008 (verlängert!) bis 13 Uhr

## 1. Aufgabe (10 Punkte)

Angegeben ist das folgende Wissensbasis (Knowledge Base):
*city*(Hildesheim), *city*(Hannover), *city*(Hamburg), *city*(Berlin), *city*(Bremen),
*way*(Hildesheim, Hannover), *way*(Hannover, Hamburg),
*way*(Hannover, Berlin), *way*(Hamburg, Brehmen)

a) [1 Punkt] Schreiben Sie auf, dass Wege nur zwischen zwei Städte führen!
b) [2 Punkte] Sie sollen das Wissensbasis mit dem Prädikat *reachable* (*erreichbar*) ergänzen. Das neue Prädikat soll zwei Klauseln haben. Eine Stadt ist erreichbar aus einer anderen, wenn es
*i)* einen unmittelbaren Weg (way) zwischen den beiden Städten gibt, bzw.
*ii)* es eine Folge von beliebig vielen Strecken gibt, so dass jede Strecke einem unmittelbaren Weg (way) entspricht.
c) [7 Punkte] Ist Hildesheim aus Bremen erreichbar? Beantworten Sie diese Frage
    i) mit Forwald Chaining und
    ii) mit Backward Chaining!
Wieviel ist die Anzahl der Schritten des Algorithmus in den beiden Fällen?

## 2. Aufgabe (7 Punkte + 3 Bonus Punkte)

a) [4 Punkte] Formulieren Sie mit den eigenen Worten, was die folgenden Begriffe bedeuten: *closed world semantic*, *universal instantiation*, *generalised modus ponens*, *unification*
b) [3 Punkte] Was sind die Prädikate, Funktionen (Funktionssymbole), Variablen und Konstanten in den folgenden Formeln:

/* hasChild($x$,$y$) = $x$ hat das Kind $y$*/
$hasChild(x, mother\_of(y))$
$hasChild(father\_of(Peter), z)$

Unter welcher Substitution sind die obigen beiden Formeln identisch? (Hinweis: Unifikationsalgorithmus)
c) [Optional, 3 Bonus Punkte] Welche der unteren Formeln folgt aus den folgenden Aussagenmenge:
$hasChild(x, mother\_of(y))$ $hasChild(Anna, x)$ $hasChild(Peter, x)$
i) $\exists w, \exists v, \exists z : hasChild(w, z), hasChild(z, v)$
ii) $\forall w, \forall v, \forall z : hasChild(w, z), hasChild(z, v)$
iii) $hasChild(Peter, Anna)$
iv) $\exists w : hasChild(Peter, w)$
v) $\forall w : hasChild(Peter, w)$

## 3. Bonus Aufgabe (10 Bonus Punkte, Abgabe 01.07.2008)

Ein Softwarezusammenstellungsystem ist in Prolog zu implementieren. Dabei sind Folgendes zu beachten:

i) Es gibt einige Software die nicht gleichzeitig installiert werden können. Beispielsweise kann *Microsoft Word 2.0 for Windows 3.1* nicht zusammen mit *Microsoft Windows Vista* installiert werden (zumindest macht es keinen Sinn).
ii) Es gibt Software, die andere Softwares benötigen. Zum Beispiel *Microsoft Word Vista* benötigt *Windows Vista* damit es richtig funktioniert.
iii) Es gibt Softwarekombinationen, die andere Software benötigen. (Also eine Menge von Software benötigt ein anderes Software, aber kein Software alleine aus der Menge benötigt das zusätzliche Software.)

Eine Menge von Software heißen wir eine Softwarezusammenstellung. Zu implementieren ist ein Program in Prolog, das entscheidet, ob eine Zusammenstellung zulässig ist, oder nicht.

In dem Wissensbasis Ihres Programmes soll es mindestens 10 Softwares geben, alle drei Arten der obigen Abhängigkeiten müssen vorkommen.

Das Hauptprogramm wird durch das Prädikat *zulaessig* realisiert. Die Softwarezusammenstellung wird durch eine Liste angegeben.

Beispiele: zulaessig([Linux, MicrosoftOffice]) → fail
zulaessig([Linux, WindowsEmulator, MicrosoftOffice] → yes

Das Source Code Ihres Programmes ist abzugeben.