Miriam Butt und Maribel Romero
G220/5109, G212/2728
{miriam.butt|maribel.romero}@uni-konstanz.de

Advanced Computational Semantics
Sommersemester 2008, Hauptseminar

Übung 1

- mit der Arbeit an den Macs vertraut machen
- swi-prolog starten und ein erstes kleines Programm schreiben
- das fertige Programm bitte an sebastian.sulger@uni-konstanz.de schicken

Aufgabe 1)

a) Melde dich mit dem Benutzer 'cluser' und dem Passwort 'spektrum' an den Rechnern an.
b) Öffne X11 mit einem Klick auf das schwarze X auf weißem Hintergrund unten in der Dock-Leiste; es öffnet sich ein Terminal, das Unix-Befehle entgegen nimmt.
d) Mit 'mkdir AdvCompSem' + Druck auf 'enter' legst du dort ein neues Verzeichnis an ('mkdir' für 'make directory').

Aufgabe 2)

a) Wechsle in das eben angelegte Verzeichnis mit 'cd AdvCompSem' + Druck auf 'enter' (du musst nicht den ganzen Namen tippen; mit der Tabulator-Taste kannst du Pfadangaben automatisch vervollständigen lassen, also z.B. 'cd Adv' und dann Druck auf 'tab' und 'enter'). Du befindest dich jetzt im neu erstellten Ordner 'AdvCompSem'.
b) Am besten, du legst dort ein extra Unterverzeichnis an mit einem Kürzel für deinen Namen: Max Mustermann legt bitte mit 'mkdir mmustermann' und 'enter' sein Verzeichnis an.
c) Gehe wiederum in deinen eben angelegten Ordner: 'cd deinKürzel'
d) Starte den emacs mit 'emacs &' + 'enter'. Der emacs ist ein Text-Editor,

der gut für alle möglichen Aufgaben benutzt werden kann, u.a. auch, um ein Prolog-Programm zu schreiben.
e) Klicke im eben geöffneten emacs-Fenster oben links auf das Symbol mit dem leeren weißen Blatt. Tippe nun 'uebung1.pl' und bestätige mit 'enter'. Du hast eine neue Datei angelegt.

Aufgabe 3)

a) Nun kannst du dein erstes Prolog-Programm schreiben. Wir fangen mit ein paar einfachen Fakten an:

```
mann(homer).
mann(apu).
junge(bart).
frau(marge).
frech(bart).
faul(homer).
reaktorunfall.
```

Es gibt 2 Männer, Homer und Apu, einen Jungen, Bart, und eine Frau, Marge. Außerdem wird gesagt, dass Bart frech ist und Homer faul und dass ein Reaktorunfall stattfindet. Speichere dein Programm: Druck auf 'ctrl' (gedrückt halten) + 'x' + 's' drücken.
b) Wir wollen diese "Wissensbasis" ('knowledge base') jetzt in swi-prolog laden und testen. Dazu wechselst du wieder in das Terminal von X11 und gibst 'swipl' ein.
c) Um das Programm zu laden: '[uebung1].' und mit 'enter' bestätigen. Die Endung '.pl' wird nicht benötigt. swi-prolog zeigt eine Meldung an, dass der Ladevorgang erfolgreich war. Du siehst an den Zeichen `?-`, das Prolog bereit ist, Befehle entgegen zu nehmen.

Aufgabe 4)

a) Wir wollen das Programm nun mit ein paar einfachen Anfragen ('queries') testen. Gültige Anfragen wären zum Beispiel:

```
mann(homer).
junge(bart).
frech(bart).
reaktorunfall.
```

swi-prolog wird auf diese Anfragen jeweils mit `true.` antworten, da diese aus der Wissensbasis als bewiesen erfasst werden können (kein Wunder, sie stehen ja genau so drin).
b) Gebe nun die folgenden Anfragen ein und überprüfe das Ergebnis:

```
junge(milhouse).
frau(edna).
```

Diese Anfragen können mit dem eingegebenen Programm von Prolog nicht als wahr bewiesen werden; Prolog weiß nichts darüber, ob Milhouse auch ein Junge ist oder Edna eine Frau, und diese Informationen können nicht aus dem Programm erschlossen werden. Daher antwortet swi-prolog mit `fail.`

c) Ein paar weitere Anfragen:

```
betrunken(homer).
pflichtbewusst(marge).
party.
```

swi-prolog zeigt eine Error-Meldung, da die sogenannten "Prädikate" `betrunken` und `pflichtbewusst` und auch die Proposition `party` Prolog nicht bekannt sind.[1]

Aufgabe 5)

a) Bis nächste Woche: rumprobieren mit Prolog! Es soll ein kleines Programm erstellt werden mit 4-6 Fakten und Propositionen. Dieses sollte in swi-prolog mit einigen Anfragen überprüft werden.

[1] Jegliche Befehle an Prolog (z.B. Fakten, Anfragen, Regeln) müssen übrigens mit einem Punkt abgeschlossen werden.