RWTH Aachen **WS 2013/14**
Lehrgebiet Theoretische Informatik Übungsblatt 13
Frohn–Reidl–Rossmanith–Sánchez–Ströder 03.02.2014

Übung zur Vorlesung Programmierung

Dieses Übungsblatt ist ein reines Bonus-Blatt. Das bedeutet, dass die Punkte auf diesem Blatt nicht zur maximal erreichbaren Punktzahl hinzu zählen, die wir zur Berechnung des prozentualen Punkteanteils verwenden. Die durch die Bearbeitung dieses Blatts gesammelten Punkte zählen wir jedoch ganz normal zu Ihren Punkten hinzu.

Aufgabe T29

Gegeben seien die folgenden Fakten und Regeln:

```
mymember(E, [E|_]).
mymember(E, [_|XS]) :- mymember(E, XS).

notmember(A,[]).
notmember(A,[B|C]) :- A\=B, notmember(A,C).

myappend([], Y, Y).
myappend([X|XS], Y, [X|XXS]) :- myappend(XS, Y, XXS).

umdrehen([], _, []).
%Regel 1:
umdrehen([X|XS], F, R) :- notmember(X, F), myappend(RR, [X], R),
   umdrehen(XS, F, RR).
% Regel 2:
umdrehen([X|XS], F, R) :- mymember(X,F), R=[X|RR], umdrehen(XS, F,
   RR).
```

Dieses Prädikat überprüft, ob Folgendes gilt:

- Sei `R1` die Liste, die entsteht, wenn aus `[X|XS]` alle Elemente gelöscht werden, die nicht in `F` vorkommen.
- Sei `R2` die Liste, die entsteht, wenn aus `[X|XS]` alle Elemente gelöscht werden, die in `F` vorkommen.
- Sei `R2'` die Liste, die entsteht, wenn `R2` umgedreht wird.
- Dann ist `R` die Liste, die entsteht, wenn `R1` und `R2'` konkateniert werden.

Es gilt also z.B. `umdrehen([1,2,3,4,5],[2],[2,5,4,3,1]).`
Zeigen Sie Schritt für Schritt die Auswertung der Anfrage

```
?- umdrehen([1,2], [2], [2,1]).
```

Lösungsvorschlag

```
% Faktum:
umdrehen([1,2], [2], [2,1]) =? umdrehen([], _, []) -> fail

% Regel 1:
umdrehen([1,2], [2], [2,1]) =? umdrehen([X1|XS1], F1, R1) -> {X1=1,
   XS1=[2], F1=[2], R1=[2,1]}
% 1. Teilziel:
   notmember(1, [2]) =? notmember(A2,[]) -> fail
   notmember(1, [2]) =? notmember(A3,[B3|C3]) -> {A3=1, B3=2, C3=[]}
      1 \= 2 -> true
      notmember(1,[]) =? notmember(A4,[]) -> {A4=1}
% 2. Teilziel:
   myappend(RR1, [1], [2,1]) =? myappend([], Y5, Y5) -> fail
   myappend(RR1, [1], [2,1]) =? myappend([X6|XS6], Y6, [X6|XXS6])
      -> {RR1=[2|XS6], Y6=[1], X6=2, XXS6=[1]}
      myappend(XS6, [1], [1]) =? myappend([], Y7, Y7) -> {XS6=[],
         Y7=[1]}

% 3. Teilziel:
% Faktum
   umdrehen([2], [2], [2]) =? umdrehen([], _, []) -> fail

% Regel 1:
   umdrehen([2], [2], [2]) =? umdrehen([X8|XS8], F8, R8) -> {X8=2,
      XS8=[], F8=[2], R8=[2]}
% 1. Teilziel:
    notmember(2, [2]) =? notmember(A9,[]) -> fail
    notmember(2, [2]) =? notmember(A10,[B10|C10]) -> {A10=2, B10=2,
       C10=[]}
       2 \= 2 -> false

% BACKTRACK
% Regel 2:
   umdrehen([2], [2], [2]) =? umdrehen([X8|XS8], F8, R8) -> {X8=2,
      XS8=[], F8=[2], R8=[2]}
% 1. Teilziel:
      mymember(2, [2]) =? mymember(E9, [E9,_]) -> {E9=2}
% 2. Teilziel:
      [2] =? [2|RR10] -> {RR10=[]}
% 3. Teilziel:
      umdrehen([], [2], [] =? umdrehen([], _, []) -> {}

Antwort: yes
```

Aufgabe T30

Die Piraten der *Neptuns Rache* haben ein Handelsschiff geplündert und wollen jetzt die reichliche Beute aufteilen. Besonders beliebt bei den Piraten sind einige Luxusgegenstände, die sie reichen Mitfahrern des Handelsschiffes abnahmen. Glücklicherweise

hat jeder Pirat einige Kombinationen von Gegenständen, mit denen er sich zufrieden geben würde—das Aufteilen erweist sich jedoch als schwierig und eine Meuterei steht kurz bevor.
Erstellen Sie ein Prolog-Programm, das zusammen mit einer Spezifikation der Form

```
piraten([john,jack,muse]).

gegenstaende([gold,pistole,diamant,diadem,truhe]).

mag(john, [gold,pistole]).
mag(john, [gold,diamant]).
mag(john, [truhe,diamant]).
mag(jack, [diamant,gold]).
mag(jack, [pistole,diamant]).
mag(jack, [pistole,diadem]).
mag(jack, [pistole,truhe]).
mag(muse, [diadem,diament]).
mag(muse, [diadem,gold]).
mag(muse, [diamant,gold]).
mag(muse, [truhe]).
```

eine Verteilung der Beute bestimmen kann, die alle Piraten zufriedenstellt und die *Neptuns Rache* vor einer blutigen Auseinandersetzung rettet!

Lösungsvorschlag

```
/* Alle Piraten */
piraten([john,jack,muse]).

/* Alle Gegenstaende */
gegenstaende([gold,pistole,diamant,diadem,truhe]).

/* Womit sind die einzelnen Piraten zufriedenzustellen?*/
mag(john, [gold,pistole]).
mag(john, [gold,diamant]).
mag(john, [truhe,diamant]).
mag(jack, [diamant,gold]).
mag(jack, [pistole,diamant]).
mag(jack, [pistole,diadem]).
mag(jack, [pistole,truhe]).
mag(muse, [diadem,diament]).
mag(muse, [diadem,gold]).
mag(muse, [diamant,gold]).
mag(muse, [truhe]).

/*****************************************************************/

/* Ist G ein Gegenstand? */
gegenstand(G) :- gegenstaende(GS), member(G, GS).

/* Ist P ein Pirat? */
```

```
pirat(P) :- piraten(PS), member(P, PS).

/* zufrieden(PS, V):
   Sind alle Piraten in der Liste PS mit der Verteilung V zufrieden?
      */
zufrieden([], V).
zufrieden([P|PS], V) :-
          mag(P, D),
          bekommt_alles(P, D, V),
          zufrieden(PS, V).

/* bekommt_alles(P, GS, V):
   Bekommt der Pirat P in der Verteilung V alle Gegenstaende in
   der Liste GS? */
bekommt_alles(P, [], _).
bekommt_alles(P, [X|D], V) :- member([P,X], V), bekommt_alles(P, D,
   V).

/* pirat_gegenstaende_paare(L1, L2):
   Sind L1 und L2 gleichlange Listen, wobei L2 aus Gegenstaenden
   besteht und L1 aus Paaren von Pirat und Gegenstand.
   Enthaelt L2 irgendwo einen Gegenstand G, dann muss L1 an derselben
   Position ein Paar [P,G] enthalten, wobei P ein Pirat ist und G
   derselbe Gegenstand.
   Beispiel:
   L1 = [diamant,gold,truhe]
   L2 = [[john,diamant],[jack,gold],[john,truhe]]
*/
pirat_gegenstaende_paare([], []).
pirat_gegenstaende_paare([[P,G]|R1], [G|R2]) :-
          pirat(P),
          gegenstand(G),
          pirat_gegenstaende_paare(R1,R2).

/* verteilung(V) sagt, dass V ein Verteilung der Gegenstaende auf
   die Piraten ausdrueckt, sodass alle Piraten zufrieden sind.
   Somit ist V eine Loesung des Raetsels.
   V ist eine Liste von Paaren [P,G], wobei P ein Pirat ist und den
   Gegenstand G erhaelt.
*/
verteilung(V) :-
          gegenstaende(L),
          pirat_gegenstaende_paare(V, L),
          piraten(PS),
          zufrieden(PS, V).

/* Permutationen (aus der Vorlesung) */
removed(X, [X|Y], Y).
removed(X, [X1|Y], [X1|Y_ohne_X]) :-
  X \= X1,
  removed(X, Y, Y_ohne_X).
```

```
perm([], []).
perm([X|Y], Z) :- member(X,Z), removed(X, Z, Z1), perm(Y, Z1).
```

Aufgabe H35 (5 Punkte)
Gegeben seien die folgenden Fakten und Regeln:

```
niedlich(pinguin).
niedlich(eichhoernchen).
niedlich(kanarienvogel).
niedlich(katze).

gefaehrlich(wolf).
gefaehrlich(orang_utan).
gefaehrlich(adler).

ungefaehrlich(X) :- X \= wolf, X \= orang_utan, X \= adler.

vierbeiner(eichhoernchen).
vierbeiner(katze).
vierbeiner(wolf).

zweibeiner(orang_utan).
zweibeiner(pinguin).
zweibeiner(kaenguru).

fliegend(kanarienvogel).
fliegend(adler).

mag(peter, X) :- niedlich(X), vierbeiner(X). % Regel 1
mag(peter, X) :- gefaehrlich(X), fliegend(X). % Regel 2
mag(peter, X) :- zweibeiner(X), ungefaehrlich(X). % Regel 3
```

Zeigen Sie Schritt für Schritt die Auswertung der Anfrage

```
?- mag(peter, orang_utan).
```

Das folgende Beispiel zeigt, wie Ihre Lösung aussehen könnte:

```
r(a,b).
q(X,_) :- r(Y,Y), q(X,Y). % Regel 1
q(X,Y) :- r(Y,X).   % Regel 2

?- q(b,X).

% Regel 1
q(b,X) =? q(X1,_) -> {X1=b}
% 1. Teilziel
   r(Y1,Y1) =? r(a,b) -> fail

% BACKTRACK
% Regel 2
q(b,X) =? q(X1,Y1) -> {X1=b, Y1=X}
```

```
% 1. Teilziel
   r(X,b) =? r(a,b) -> {X=a}

Antwort: {X=a}
```

Lösungsvorschlag

```
% Regel 1
mag(peter, orang_utan) =? mag(peter, X1) -> {X1=orang_utan}
% 1. Teilziel
   niedlich(orang_utan) =? niedlich(pinguin) -> fail
   niedlich(orang_utan) =? niedlich(eichhoernchen) -> fail
   niedlich(orang_utan) =? niedlich((kanarienvogel) -> fail
   niedlich(orang_utan) =? niedlich(katze) -> fail

% BACKTRACK
% Regel 2
mag(peter, orang_utan) =? mag(peter, X1) -> {X1=orang_utan}
% 1. Teilziel
   gefaehrlich(orang_utan) =? gefaehrlich(wolf) -> fail
   gefaehrlich(orang_utan) =? gefaehrlich(orang_utan) -> {}
% 2. Teilziel
   fliegend(orang_utan) =? fliegend(kanarienvogel) -> fail
   fliegend(orang_utan) =? fliegend(adler) -> fail

% BACKTRACK
% Regel 3
mag(peter, orang_utan) =? mag(peter, X1) -> {X1=orang_utan}
% 1. Teilziel
   zweibeiner(orang_utan) =? zweibeiner(orang_utan) -> {}
% 2. Teilziel
   ungefaehrlich(orang_utan) =? ungefaehrlich(X) -> {X=orang_utan}.
   % 1. Teilziel
      orang_utan \= wolf -> true
   % 2. Teilziel
      orang_utan \= orang_utan -> false


Antwort: no.
```

Aufgabe H36 (10 Punkte)

An einem kühlen See im staubigen New Mexico finden sich fünf Ranches. In jeder Ranch wohnt ein Cowboy; jeder dieser Cowboys kommt aus einem anderen US-Bundesstaat und hat seiner Ranch einen persönlichen Namen gegeben. Außerdem hat jeder Cowboy ein Lieblingsessen, einen Lieblingsdrink und besitzt eine Waffe. Diese Sachverhalte seien durch folgende Prologprädikate ausgedrückt:

```
herkunft(montana).
herkunft(nevada).
```

```
herkunft(south_carolina).
herkunft(vermont).
herkunft(california).

ranch(ponderosa).
ranch(san_antonio).
ranch(mary_magdalene).
ranch(lillypond).
ranch(the_swamp).

essen(pferd).
essen(bohnen).
essen(fajitas).
essen(buffalo).
essen(souffle).

waffe(messer).
waffe(pistole).
waffe(shotgun).
waffe(feder).
waffe(intelligenz).

drink(moonshine).
drink(whisky).
drink(gin).
drink(kamillentee).
drink(aguardiente).
cowboy(Ranch, Staat, Waffe, Drink, Essen) :-
     ranch(Ranch), herkunft(Staat),
     drink(Drink), waffe(Waffe),
     essen(Essen).
```

Die fünf Ranches befinden sich alle in einer Reihe am Seeufer. Folgende Fakten sind über die Ranches und ihre Besitzer bekannt:

- Der Cowboy, der in der Ponderosa'-Ranch wohnt, ist aus Californien.
- Der Cowboy aus Vermont isst am liebsten Büffel.
- Die Waffe des Cowboys aus South Carolina ist seine Intelligenz.
- Die Ranch 'The Swamp' ist links von 'Mary Magdalene'.
- Die Waffe des Cowboys in 'The Swamp' ist eine Feder.
- Der Cowboy, der am liebsten Fajitas isst, trinkt am liebsten Aguardiente.
- Der Cowboy in der mittleren Ranch hat eine Schrotflinte.
- Der Cowboy im 'Lillypond' trinkt Kamillentee.
- Der Cowboy im linkesten Haus ist aus Montana.
- Der Cowboy, der Moonshine trinkt, wohnt neben dem Cowboy, der Bohnen isst.

- Der Cowboy, der Pferd isst, wohnt neben dem Kamillentee trinkenden Cowboy.
- Der Cowboy, der ein Messer besitzt, trinkt Whisky.
- Die Ranch des Cowboys aus Montana ist neben der 'San Antonio' Ranch.
- Der Cowboy aus Nevada trinkt Gin.
- Der Cowboy, der Moonshine trinkt, wohnt neben dem Cowboy mit der Pistole.

Der folgende Code sei dazu vorgegeben:

```
mittleres(M,[_,_,M,_,_]).

solution(N) :-
     Ranches = [_,_,_,_,_],
     member(cowboy(ponderosa,california,_,_,_),Ranches),
     ...
```

Schreiben Sie die Prädikate `links`, `erstes` und `neben` und vervollständigen Sie das Prädikat `solution`, so dass man damit folgende Frage beantworten kann: Aus welchem Staat ist der Cowboy, der am liebsten Soufflé isst?

Lösungsvorschlag

```
erstes(E,[E|_]). mittleres(M,[_,_,M,_,_]).
links(A,B,[A,B|_]). links(A,B,[_|R]) :- links(A,B,R).
neben(A,B,L) :- links(A,B,L).
neben(A,B,L) :- links(B,A,L).

solution(N) :-
  Ranches = [_,_,_,_,_],
  member(cowboy(ponderosa,california,_,_,_),Ranches),
  member(cowboy(_,vermont,_,_,buffalo),Ranches),
  member(cowboy(_,south_carolina,intelligenz,_,_),Ranches),
  links(cowboy(the_swamp,_,_,_,_),cowboy(mary_magdalene,_,_,_,_),Ranches),
  member(cowboy(the_swamp,_,feder,_,_),Ranches),
  member(cowboy(_,_,_,aguardiente,fajitas),Ranches),
  mittleres(cowboy(_,_,shotgun,_,_),Ranches),
  member(cowboy(lillypond,_,_,kamillentee,_),Ranches),
  erstes(cowboy(_,montana,_,_,_),Ranches),
  neben(cowboy(_,_,_,moonshine,_),cowboy(_,_,_,_,bohnen),Ranches),
  neben(cowboy(_,_,_,_,pferd),cowboy(_,_,_,kamillentee,_),Ranches),
  member(cowboy(_,_,messer,whisky,_),Ranches),
  neben(cowboy(_,montana,_,_,_),cowboy(san_antonio,_,_,_,_),Ranches),
  member(cowboy(_,nevada,_,gin,_),Ranches),
  neben(cowboy(_,_,_,moonshine,_),cowboy(_,_,pistole,_,_),Ranches),
  member(cowboy(_,N,_,_,souffle),Ranches).
```

Aufgabe H37 (10 Punkte)

Vier Maschinenbauer stehen in einer Reihe (von links nach rechts) vor dem Audimax. Jeder Maschi trägt ein anderes Hemd, einer von Ihnen ein rotes. Der Maschi direkt rechts von Rafael trägt eine blaues Hemd. Harald ist der zweite in der Reihe. Bob trägt das klassische Karohemd. Thomas ist nicht der vierte, und er trägt kein graues Hemd.

In welcher Reihenfolge stehen die vier Maschis in der Reihe und wer trägt welches Hemd?

Entwerfen Sie ein Programm in Prolog, welches das obige Rätsel löst!

Lösungsvorschlag

```
hemdfarbe(blau).
hemdfarbe(karo).
hemdfarbe(grau).
hemdfarbe(rot).

position(1).
position(2).
position(3).
position(4).

notmember(A,[]).
notmember(A,[B|C]) :- A\=B, notmember(A,C).

plus(A, B, C) :- C is A + B.

ist_menge([]).
ist_menge([_]).
ist_menge([A|AS]) :- notmember(A,AS), ist_menge(AS).

maschi( [_, C, P] ) :- hemdfarbe(C), position(P).

maschiRaetsel( G1, G2, G3, G4) :-
    maschi(G1),maschi(G2),maschi(G3), maschi( G4),
    G1 = [rafael, RafaelFarbe, RafaelPosition ],
    G2 = [harald, HaraldFarbe, 2],
    G3 = [thomas, ThomasFarbe, ThomasPosition],
    ThomasPosition \= 1, ThomasPosition \= 4, ThomasFarbe \= grau,
    G4 = [bob, karo, BobPosition],
    ist_menge([RafaelFarbe, HaraldFarbe, ThomasFarbe, karo]),
    ist_menge([RafaelPosition, ThomasPosition, BobPosition, 2]),
    plus( RafaelPosition, 1, FP1),
    member( [_, blau, FP1], [G1, G2, G3, G4]).
```

Abgabe zum 11.02.2014