Logisches Schließen in Computerprogramme umsetzen

Nikolay Paleshnikov

Logik zwischen Mathematik und Philosophie

Göttingen

30. April 2017

Logische Schlussfolgerung

(1) Bedingung $\xrightarrow{(2)\text{Regel}}$ (3) Konsequenz, e.g.

(1) Der Himmel ist wolkenbedeckt.
(2) Bei wolkenbedecktem Himmel regnet es.
(3) Es regnet.

Logische Schlussfolgerung

(1) Bedingung $\xrightarrow{(2)\text{Regel}}$ (3) Konsequenz, e.g.

(1) Der Himmel ist wolkenbedeckt.
(2) Bei wolkenbedecktem Himmel regnet es.
(3) Es regnet.

- Deduktion
  gegeben (1) und (2), folgere (3)
- Abduktion
  gegeben (2) und (3), finde geeignetes (1)
- Induktion
  gegeben (1) und (3), finde geeignetes (2)

Deduktionsbeispiel in Prolog

Es sei das folgende Rätsel gegeben

Die drei Jungen Fritz, Hans und Karl rudern mit ihren Booten auf einem See. Die Boote haben die Farben rot, grün und blau. Der Junge im roten Boot ist der Bruder von Fritz, Hans sitzt nicht im grünen Boot, der Junge im grünen Boot hat Streit mit Fritz. Wir suchen jeweils den Namen des Jungen im roten, grünen bzw. blauen Boot.

Deduktionsbeispiel in Prolog

Es sei das folgende Rätsel gegeben

Die drei Jungen Fritz, Hans und Karl rudern mit ihren Booten auf einem See. Die Boote haben die Farben rot, grün und blau. Der Junge im roten Boot ist der Bruder von Fritz, Hans sitzt nicht im grünen Boot, der Junge im grünen Boot hat Streit mit Fritz. Wir suchen jeweils den Namen des Jungen im roten, grünen bzw. blauen Boot.

Das von Prolog zu lösende Programm ist dann

```
person(fritz). person(hans). person(karl).
farbe(rot). farbe(gruen). farbe(blau).
boot(N,F):-person(N),farbe(F).
zuteilung(HF,FF,KF):-
    boot(hans,HF), HF\=gruen,
    boot(fritz,FF),FF\=rot, FF\=gruen,HF\=FF,
    boot(karl,KF), KF\=HF, KF\=FF.
```

Deduktionsbeispiel in Prolog

Es sei das folgende Rätsel gegeben

Die drei Jungen Fritz, Hans und Karl rudern mit ihren Booten auf einem See. Die Boote haben die Farben rot, grün und blau. Der Junge im roten Boot ist der Bruder von Fritz, Hans sitzt nicht im grünen Boot, der Junge im grünen Boot hat Streit mit Fritz. Wir suchen jeweils den Namen des Jungen im roten, grünen bzw. blauen Boot.

Das von Prolog zu lösende Programm ist dann

```
person(fritz). person(hans). person(karl).
farbe(rot). farbe(gruen). farbe(blau).
boot(N,F):-person(N),farbe(F).
zuteilung(HF,FF,KF):-
    boot(hans,HF), HF\=gruen,
    boot(fritz,FF),FF\=rot, FF\=gruen,HF\=FF,
    boot(karl,KF), KF\=HF, KF\=FF.
```

?-zuteilung(HF,FF,KF). HF = rot, FF = blau, KF = gruen

Herbrand-Universum und Herbrand-Modell

- Die schwache Negation *not p* wird als wahr ausgewertet g.d.w. p nicht durch Resolution bewiesen werden kann.
- Die Menge aller Grundatome eines Logikprogrammes P wird als ihr Herbrand-Universum $HB(P)$ bezeichnet.

Herbrand-Universum und Herbrand-Modell

- Die schwache Negation *not p* wird als wahr ausgewertet g.d.w. p nicht durch Resolution bewiesen werden kann.
- Die Menge aller Grundatome eines Logikprogrammes P wird als ihr Herbrand-Universum $HB(P)$ bezeichnet.

Für das logische Programm Q

$p(X) \leftarrow \mathbf{not}\ q(X), r(X).$
$q(X) \leftarrow r(X), s.$
$s \leftarrow \mathbf{not}\ t.$
$\leftarrow s, t.$
$r(a).$

gilt $HB(Q) = \{s, t, p(a), q(a), r(a)\}$.

Herbrand-Universum und Herbrand-Modell

- Die schwache Negation *not p* wird als wahr ausgewertet g.d.w. p nicht durch Resolution bewiesen werden kann.
- Die Menge aller Grundatome eines Logikprogrammes P wird als ihr Herbrand-Universum $HB(P)$ bezeichnet.

Für das logische Programm Q

$p(X) \leftarrow \mathbf{not}\ q(X), r(X).$
$q(X) \leftarrow r(X), s.$
$s \leftarrow \mathbf{not}\ t.$
$\leftarrow s, t.$
$r(a).$

gilt $HB(Q) = \{s, t, p(a), q(a), r(a)\}$.

- Eine Herbrand-Interpretation $\mathfrak{I}$ von P ist eine Teilmenge von $HB(P)$, die jedem Grundatom in $HB(P)$ ein Wahrheitswert zuweist.
- Ein Herbrand-Modell M von P ist eine Herbrand-Interpretation, die jede Klausel von P erfüllt.

Herbrand-Universum und Herbrand-Modell

- Die schwache Negation *not p* wird als wahr ausgewertet g.d.w. p nicht durch Resolution bewiesen werden kann.
- Die Menge aller Grundatome eines Logikprogrammes P wird als ihr Herbrand-Universum $HB(P)$ bezeichnet.

Für das logische Programm Q

$p(X) \leftarrow \mathbf{not}\ q(X), r(X).$
$q(X) \leftarrow r(X), s.$
$s \leftarrow \mathbf{not}\ t.$
$\leftarrow s, t.$
$r(a).$

gilt $HB(Q) = \{s, t, p(a), q(a), r(a)\}$.

- Eine Herbrand-Interpretation $\mathfrak{I}$ von P ist eine Teilmenge von $HB(P)$, die jedem Grundatom in $HB(P)$ ein Wahrheitswert zuweist.
- Ein Herbrand-Modell M von P ist eine Herbrand-Interpretation, die jede Klausel von P erfüllt.

{q(a), r(a), s} und {p(a), r(a), t} sind Herbrand-Modelle von Q. Sie sind beide minimal, da keine ihrer echten Teilmengen ein Modell von Q ist.

Answer Sets

Seien P ein über Grundatome definiertes Logikprogramm und X eine Menge von Grundatomen. Das Redukt P^X erhält man durch Löschen aller Regeln, in deren Rumpf negative Literale über Atome aus X vorkommen, und aller negativen Literale aus den Rümpfen der übriggebliebenen Regeln.

Answer Sets

Seien P ein über Grundatome definiertes Logikprogramm und X eine Menge von Grundatomen. Das Redukt P^X erhält man durch Löschen aller Regeln, in deren Rumpf negative Literale über Atome aus X vorkommen, und aller negativen Literale aus den Rümpfen der übriggebliebenen Regeln.

Die Menge X ist ein Answer set (AS) vom Logikprogramm P gdw. X ist der einzige minimale Herbrand-Modell von P^X.

Beispiel zu Answer Sets

Finde die Answer sets des Programms P:

$a \leftarrow a.$
$b \leftarrow \ \mathbf{not}\ a.$

Beispiel zu Answer Sets

Finde die Answer sets des Programms P:

$a \leftarrow a.$
$b \leftarrow \ \mathbf{not}\ a.$

Bilde dazu die Redukte von P in Bezug auf jede Teilmenge von HB(P):

$P^{\emptyset} = \{a \leftarrow a.\ b.\}$ mit minimalem Herbrand-Modell $\{b\} \neq \{\emptyset\}$
$P^{\{a\}} = \{a \leftarrow a.\}$ mit minimalem Herbrand-Modell $\{\emptyset\} \neq \{a\}$
$P^{\{b\}} = \{a \leftarrow a.\ b.\}$ mit minimalem Herbrand-Modell $\{b\}$
$P^{\{a,b\}} = \{a \leftarrow a.\}$mit minimalem Herbrand-Modell $\{\emptyset\} \neq \{a, b\}$

Beispiel zu Answer Sets

Finde die Answer sets des Programms P:

$a \leftarrow a.$
$b \leftarrow \ \mathbf{not}\ a.$

Bilde dazu die Redukte von P in Bezug auf jede Teilmenge von HB(P):

$P^{\emptyset} = \{a \leftarrow a.\ b.\}$ mit minimalem Herbrand-Modell $\{b\} \neq \{\emptyset\}$
$P^{\{a\}} = \{a \leftarrow a.\}$ mit minimalem Herbrand-Modell $\{\emptyset\} \neq \{a\}$
$P^{\{b\}} = \{a \leftarrow a.\ b.\}$ mit minimalem Herbrand-Modell $\{b\}$
$P^{\{a,b\}} = \{a \leftarrow a.\}$mit minimalem Herbrand-Modell $\{\emptyset\} \neq \{a, b\}$

Somit ist $\{b\}$ allein ein AS von P.

Abduktion im Kontext von Answer Set Programming

Eine abduktive Lernaufgabe $< B, Ab, Ob >$ ist definiert durch

- Hintergrundwissen B
- Menge abduzierbarer Atome $Ab \subseteq HB(B)$
- Beobachtungen Ob, bestehend aus positiven (E^+) und negativen Beispielen (E^-)

Abduktion im Kontext von Answer Set Programming

Eine abduktive Lernaufgabe $< B, Ab, Ob >$ ist definiert durch

- Hintergrundwissen B
- Menge abduzierbarer Atome $Ab \subseteq HB(B)$
- Beobachtungen Ob, bestehend aus positiven (E^+) und negativen Beispielen (E^-)

Eine abduktive Lösung Δ of $< B, Ab, Ob >$ muss folgende Bedingungen erfüllen

- $\Delta \subseteq Ab$
- $\Delta \cup B \models Ob$
- $\Delta \cup B \not\models \bot$

Abduktionsbeispiel in Clingo

Löse die abduktive Lernaufgabe mit

- $B = \{p :- q, r.\ p :- s\}$
- $Ab = \{q, r, s\}$
- $E^+ = \{p\}$ and $E^- = \{s\}$

Abduktionsbeispiel in Clingo

Löse die abduktive Lernaufgabe mit

- $B = \{p :- q, r.\ p :- s\}$
- $Ab = \{q, r, s\}$
- $E^+ = \{p\}$ and $E^- = \{s\}$

Das entsprechende von Clingo zu lösende Programm ist das Folgende

```
p :- q,r.
p :- s.
{q,r,s}.
goal :- p, not s.
:- not goal.
```

Abduktionsbeispiel in Clingo

Löse die abduktive Lernaufgabe mit

- $B = \{p :- q, r.\ p :- s\}$
- $Ab = \{q, r, s\}$
- $E^+ = \{p\}$ and $E^- = \{s\}$

Das entsprechende von Clingo zu lösende Programm ist das Folgende

```
p :- q,r.
p :- s.
{q,r,s}.
goal :- p, not s.
:- not goal.
```

Sein einziges AS ist {q,r,p,goal}, deswegen erhält man als abduktive Lösung $\{q,r,p,goal\} \cap Ab = \{q,r\}$.

Induktion in der Induktiven Logischen Programmierung

Eine induktive Lernaufgabe $< B, E, M >$ ist definiert durch:

- Hintergrundwissen B
- positive E^+ und negative Beispiele E^-
- Kopf- M_h und Rumpfdeklarationen M_b

Induktion in der Induktiven Logischen Programmierung

Eine induktive Lernaufgabe $< B, E, M >$ ist definiert durch:

- Hintergrundwissen B
- positive E^+ und negative Beispiele E^-
- Kopf- M_h und Rumpfdeklarationen M_b

Eine induktive Lösung H of $< B, E, M >$, genannt Hypothese, muss folgende Bedingungen erfüllen:

- $H \subseteq S_M$
- $\exists a \in AS(B \cup H)\ s.t.\ \forall e \in E^+\ (e \in a) \wedge \forall e \in E^-\ (e \notin a)$

Induktionsbeispiel in ASPAL

Löse die induktive Lernaufgabe mit

- $B = \{r(a).\ t(a).\ t(b).\}$
- $M_h = \{modeh(p(+t)).\}$ and
  $M_b = \{modeb(q(+t)).\ modeb(r(+t)).\}$
- $E^+ = \{p(a)\}$ and $E^- = \{p(b)\}$

Induktionsbeispiel in ASPAL

Löse die induktive Lernaufgabe mit

- $B = \{r(a).\ t(a).\ t(b).\}$
- $M_h = \{modeh(p(+t)).\}$ and
  $M_b = \{modeb(q(+t)).\ modeb(r(+t)).\}$
- $E^+ = \{p(a)\}$ and $E^- = \{p(b)\}$

Das zugehörige Regelgerüst S_M ist

$p(A) \leftarrow t(A).$
$p(A) \leftarrow t(A), q(A).$
$p(A) \leftarrow t(A), r(A).$
$p(A) \leftarrow t(A), q(A), r(A).$

Induktionsbeispiel in ASPAL

Löse die induktive Lernaufgabe mit

- $B = \{r(a).\ t(a).\ t(b).\}$
- $M_h = \{modeh(p(+t)).\}$ and
  $M_b = \{modeb(q(+t)).\ modeb(r(+t)).\}$
- $E^+ = \{p(a)\}$ and $E^- = \{p(b)\}$

Zur Umwandlung der gegebenen induktiven in eine äquivalente abduktive Lernaufgabe erweitert man S_M zu

$p(A) \leftarrow t(A), meta(r_1).$
$p(A) \leftarrow t(A), q(A), meta(r_2).$
$p(A) \leftarrow t(A), r(A), meta(r_3).$
$p(A) \leftarrow t(A), q(A), r(A), meta(r_4).$

$\{\text{meta}(r_1), \text{meta}(r_2), \text{meta}(r_3), \text{meta}(r_4)\}.$

Induktionsbeispiel in ASPAL

Löse die induktive Lernaufgabe mit

- $B = \{r(a).\ t(a).\ t(b).\}$
- $M_h = \{modeh(p(+t)).\}$ and
  $M_b = \{modeb(q(+t)).\ modeb(r(+t)).\}$
- $E^+ = \{p(a)\}$ and $E^- = \{p(b)\}$

Zur Umwandlung der gegebenen induktiven in eine äquivalente abduktive Lernaufgabe erweitert man S_M zu

$p(A) \leftarrow t(A), meta(r_1).$
$p(A) \leftarrow t(A), q(A), meta(r_2).$
$p(A) \leftarrow t(A), r(A), meta(r_3).$
$p(A) \leftarrow t(A), q(A), r(A), meta(r_4).$

$\{\text{meta}(r_1), \text{meta}(r_2), \text{meta}(r_3), \text{meta}(r_4)\}.$

Als einzige optimale Lösung H wird

$p(A) : -t(A), r(A).$

ermittelt mit $\{r(a), t(a), t(b), p(a)\} \in AS(B \cup H)$

Induktionsbeispiel aus der echten Welt

```
% mode declarations
modeh(scientist(+child)).
modeh(explorer(+child)).
modeh(proud(+parent)).
modeh(lonely(+childless)).

modeb(scientist(+child)).
modeb(explorer(+child)).
modeb(adventurous(+child)).
modeb(curious(+child)).
modeb(offspring(+parent,
                -child)).
```

```
% background knowledge
humanist(X) :- scientist(X).
child(annika).
child(tommy).
child(jack).
parent(john).
parent(clara).
childless(persephone).
offspring(clara,annika).
offspring(john,jack).
adventurous(tommy).
curious(annika).
```

```
% examples
example(humanist(tommy),-1).
example(humanist(annika),1).
example(explorer(tommy),1).
example(explorer(annika),-1).
example(proud(clara),1).
example(proud(john),-1).
example(lonely(persephone),1).
```

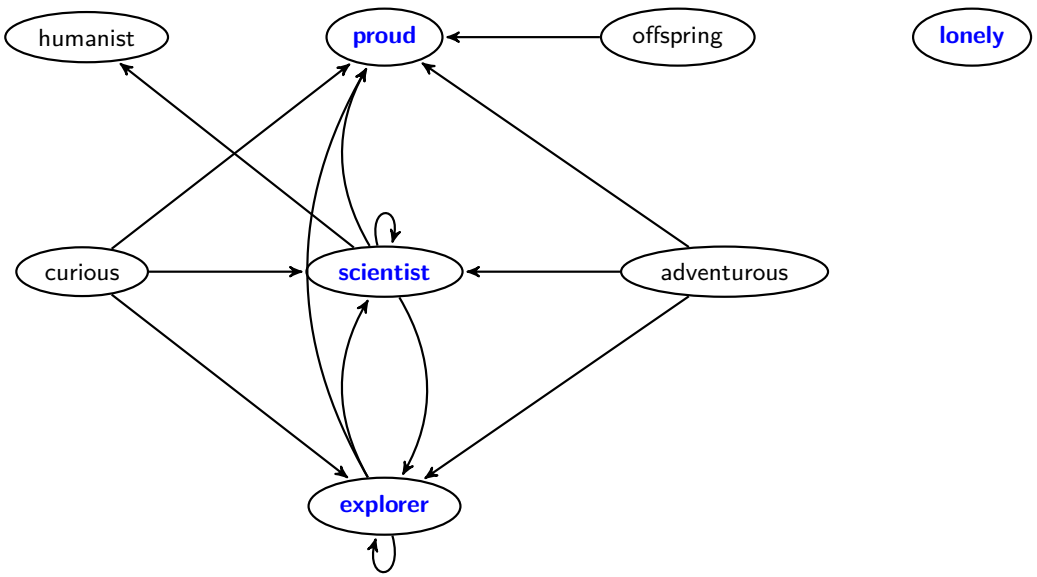
Induktionsbeispiel aus der echten Welt

```
% mode declarations
modeh(scientist(+child)).
modeh(explorer(+child)).
modeh(proud(+parent)).
modeh(lonely(+childless)).

modeb(scientist(+child)).
modeb(explorer(+child)).
modeb(adventurous(+child)).
modeb(curious(+child)).
modeb(offspring(+parent,
                -child)).
```

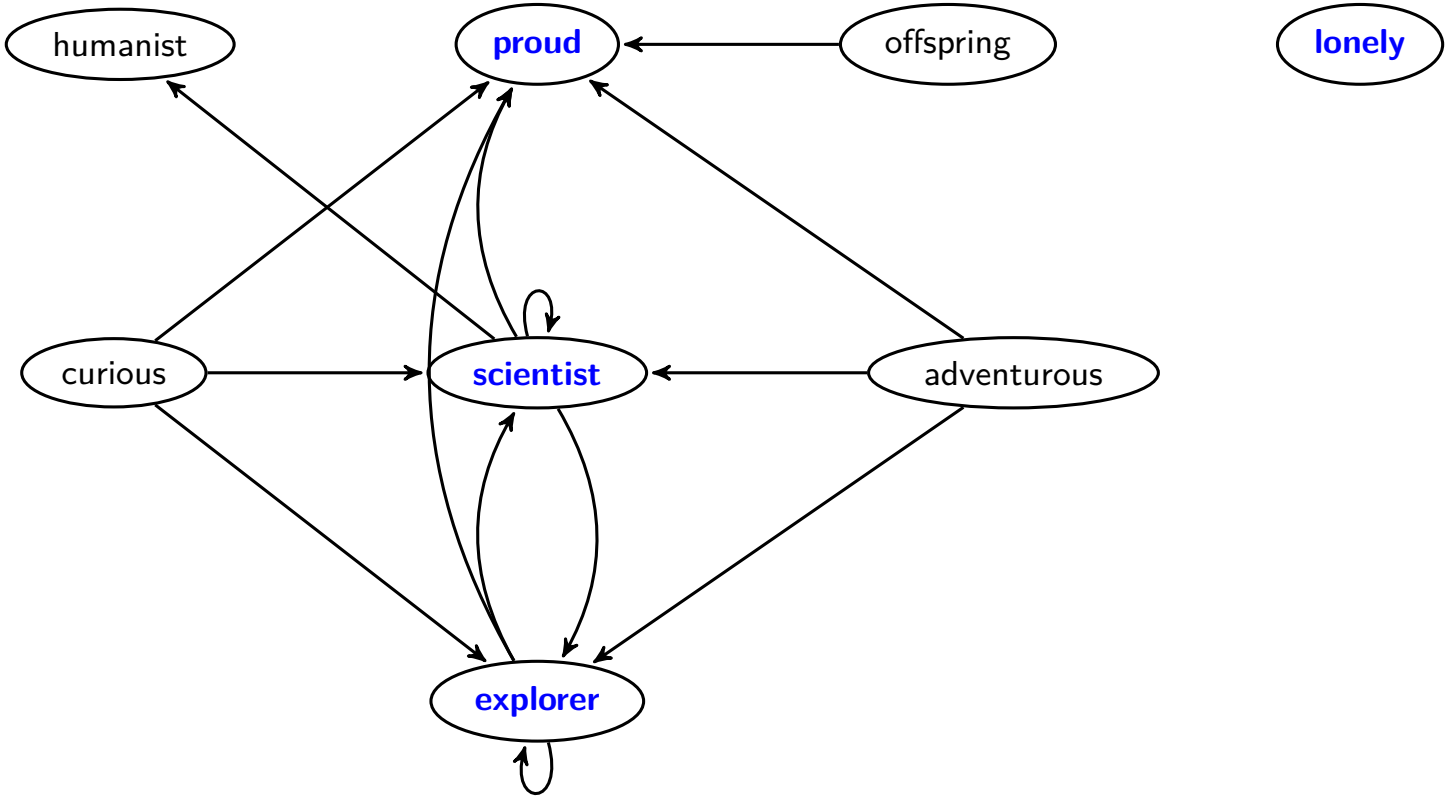
```
% background knowledge
humanist(X) :- scientist(X).
child(annika).
child(tommy).
child(jack).
parent(john).
parent(clara).
childless(persephone).
offspring(clara,annika).
offspring(john,jack).
adventurous(tommy).
curious(annika).
```

```
% examples
example(humanist(tommy),-1).
example(humanist(annika),1).
example(explorer(tommy),1).
example(explorer(annika),-1).
example(proud(clara),1).
example(proud(john),-1).
example(lonely(persephone),1).
```



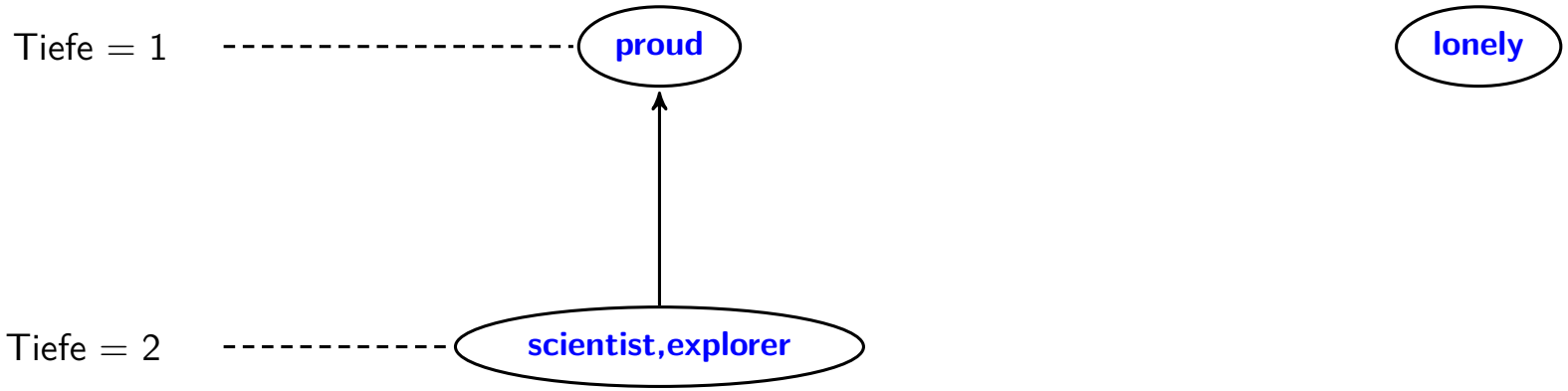
Die Prädikate aus den Kopfdeklarationen sind blau hinterlegt.

Abhängigkeitsgraph



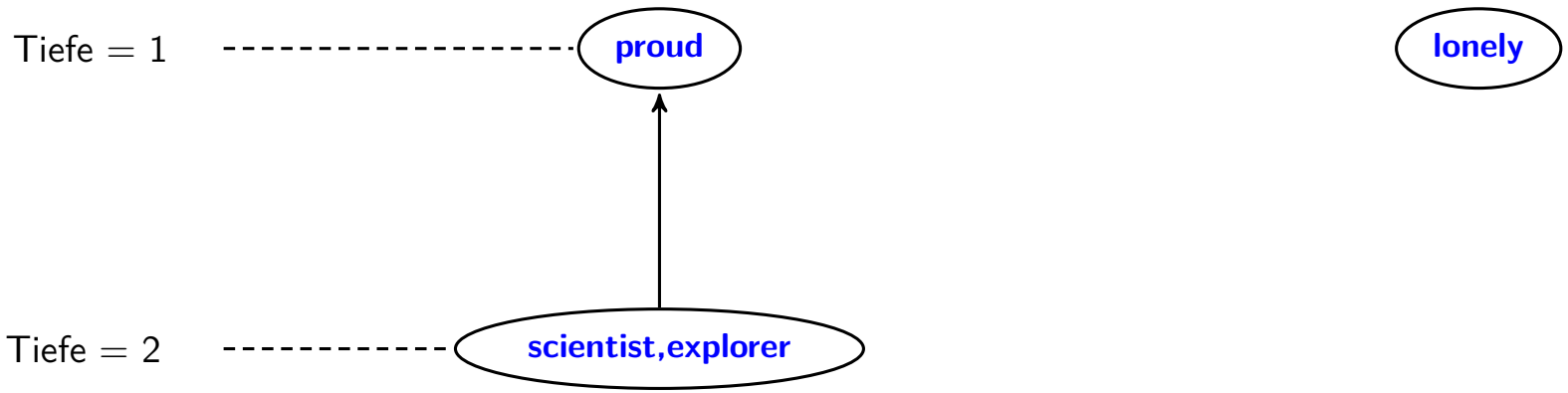
Induktive Lösungen

Reduktion des Abhängigkeitsgraphen in Bezug auf die Kopfdeklarationen



Induktive Lösungen

Reduktion des Abhängigkeitsgraphen in Bezug auf die Kopfdeklarationen




```
proud(A):-parent(A),offspring(A,B),
   child(B),scientist(B).
```

und

```
proud(A):-parent(A),offspring(A,B),
   child(B),curious(B).
```

```
lonely(A):-childless(A).
```

```
scientist(A):-child(A),curious(A).
explorer(A):-child(A),adventurous(A).
```

Induktive Lösungen


```
proud(A):-parent(A),offspring(A,B),
   child(B),scientist(B).
scientist(A):-child(A),curious(A).
explorer(A):-child(A),adventurous(A).
```

und

```
proud(A):-parent(A),offspring(A,B),
   child(B),curious(B).
scientist(A):-child(A),curious(A).
explorer(A):-child(A),adventurous(A).
```

```
lonely(A):-childless(A).
```

Induktive Lösungen

```
% mode declarations
modeh(scientist(+child)).
modeh(explorer(+child)).
modeh(proud(+parent)).
modeh(lonely(+childless)).

modeb(scientist(+child)).
modeb(explorer(+child)).
modeb(adventurous(+child)).
modeb(curious(+child)).
modeb(offspring(+parent,
                -child)).
```

```
% background knowledge
humanist(X) :- scientist(X).
child(annika).
child(tommy).
child(jack).
parent(john).
parent(clara).
childless(persephone).
offspring(clara,annika).
offspring(john,jack).
adventurous(tommy).
curious(annika).
```

```
% examples
example(humanist(tommy),-1).
example(humanist(annika),1).
example(explorer(tommy),1).
example(explorer(annika),-1).
example(proud(clara),1).
example(proud(john),-1).
example(lonely(persephone),1).
```

```
lonely(A):-childless(A).
proud(A):-parent(A),offspring(A,B),
  child(B),scientist(B
scientist(A):-child(A),curious(A).
explorer(A):-child(A),adventurous(A).
```

und

```
lonely(A):-childless(A).
proud(A):-parent(A),offspring(A,B),
  child(B),curious(B).
scientist(A):-child(A),curious(A).
explorer(A):-child(A),adventurous(A).
```

Interesse an logischer Programmierung geweckt?

Prolog

`http://www.swi-prolog.org/`

Clingo

`https://potassco.org/`

Clingo online ausführen
`http://potassco.sourceforge.net/clingo.html`

ASPAL

Code ist öffentlich und als Teil meines Abschlussprojektes zur Parallelisierung von ASPAL verfügbar
`https://github.com/Nick36/parallelizing-ILP-in-ASP`

Eine ausführlichere Behandlung von ASP und ILP findet man im zweiten Abschnitt der Ausarbeitung unter obigem Link und in der Vorlesung „Logic-Based Learning“ am Imperial College London `http://wp.doc.ic.ac.uk/arusso/myteaching/logic-based-learning/`