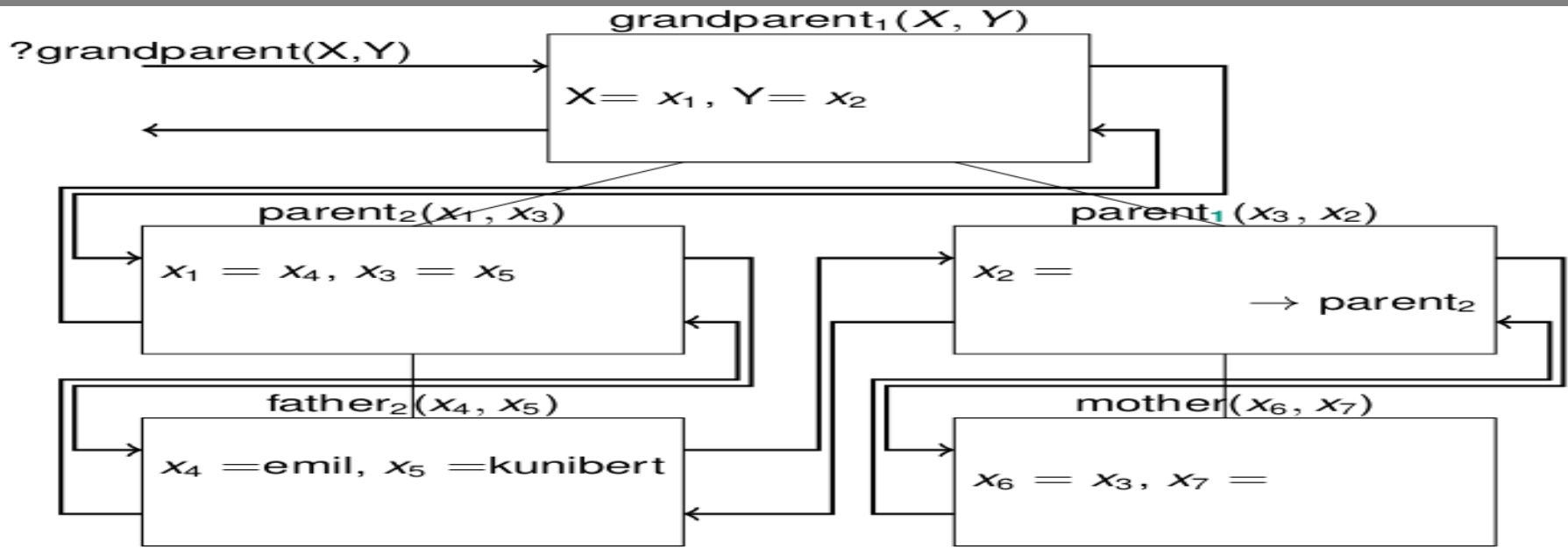
Karlsruher Institut für Technologie

Einführung in Prolog

Simon Bischof

IPD Snelting



Terme

id := [a-z][a-zA-Z0-9]*
var := [A-Z][a-zA-Z0-9]*
num := [0-9]*
Term := *Functor* | **var** | **num**
Functor := **id** | **id** **(** *Term* { **,** *Term* }* **)**

- Beispiele:

  lisa, leaf, X, node(leaf,leaf), node(X,node(Y,leaf)), 0, plus(0,3)

- Variablenfreie Terme stehen nur für sich selbst!

Regeln, Fakten

Rule := *Functor* **.** | *Functor* **:-** *Goal* { **,** *Goal* }* **.**
Goal := *Functor* | *Term* **=** *Term* | *Term* **is** *Term* | **!**

- Ohne Ziele (Fakt) = wahre Aussage
- Mit Zielen = Implikation
- Mehrere Regeln = Disjunktion
- Mehrere Ziele = Konjunktion
- Abfrage durch Eingabe von Zielliste

Fakten und Regeln

```
% Marge ist Mutter von Lisa und Bart
mother(marge, lisa).
mother(marge, bart).
father(homer, bart).
father(grampa, homer).

% Wenn Homer der Vater von Bart ist,
% ist er auch ein Elternteil von ihm
parent(homer, bart) :- father(homer, bart).
parent(grampa, homer) :- mother(grampa, homer).
parent(grampa, homer) :- father(grampa, homer).
parent(homer, nelson) :- father(homer, nelson).

% Grampa ist Großelternteil von Bart,
% wenn er Elternteil von Homer
% und Homer Elternteil von Bart ist
grandparent(grampa, bart) :-
    parent(grampa, homer), parent(homer, bart).
grandparent(grampa, nelson) :-
    parent(grampa, homer), parent(homer, nelson).
```

Variablen

- Nicht sinnvoll/möglich alle Kombinationen aufzuschreiben
- Lösung: Regeln mit Variablen
- Variablen können für beliebigen Term stehen

Variablen

- Nicht sinnvoll/möglich alle Kombinationen aufzuschreiben
- Lösung: Regeln mit Variablen
- Variablen können für beliebigen Term stehen

```
mother(marge, lisa). mother(marge, bart).
father(homer, bart). father(grampa, homer).

parent(X, Y) :- father(X, Y).
parent(X, Y) :- mother(X, Y).

grandparent(X, Y) :- parent(X, Z), parent(Z, Y).
```

Variablen

- Nicht sinnvoll/möglich alle Kombinationen aufzuschreiben
- Lösung: Regeln mit Variablen
- Variablen können für beliebigen Term stehen

```
mother(marge, lisa). mother(marge, bart).
father(homer, bart). father(grampa, homer).

parent(X, Y) :- father(X, Y).
parent(X, Y) :- mother(X, Y).

grandparent(X, Y) :- parent(X, Z), parent(Z, Y).
```

Mögliche Abfragen:

- ? mother(marge, lisa). $\Longrightarrow$ yes.
- ? mother(marge, nelson). $\Longrightarrow$ no.
- ? mother(marge, X). $\Longrightarrow$ X = lisa ; X = bart; no.
- ? grandparent(X, Y). $\Longrightarrow$ X = grampa, Y = bart; no.

Substitution

- Ersetzen von Variablen durch Terme
- Kann partielle Abbildung sein

Substitution

- Ersetzen von Variablen durch Terme
- Kann partielle Abbildung sein
- $\sigma = [X \Rightarrow b]$:
  $\sigma(f(X)) = f(b)$

Substitution

- Ersetzen von Variablen durch Terme
- Kann partielle Abbildung sein
- $\sigma = [X \Rightarrow b]$:
  $\sigma(f(X)) = f(b)$
- $\sigma = [X \Rightarrow b, Y \Rightarrow g(a)]$:
  $\sigma(f(X, Y)) = f(b, g(a))$

Substitution

- Ersetzen von Variablen durch Terme
- Kann partielle Abbildung sein
- $\sigma = [X \Rightarrow b]$:
  $\sigma(f(X)) = f(b)$
- $\sigma = [X \Rightarrow b, Y \Rightarrow g(a)]$:
  $\sigma(f(X, Y)) = f(b, g(a))$
- $\sigma = [X \Rightarrow b, Z \Rightarrow W, A \Rightarrow f(a, Y)]$:
  $\sigma(g(X, Y, f(X, W), h(Z), f(A, B))) =$
  $g(b, Y, f(b, W), h(W), f(f(a, Y), B))$

Unifikation

- Gegeben: Menge von Termgleichungen
- Gesucht: Substitution, nach deren Anwendung alle Gleichungen erfüllt sind (Unifikator)
- $\{X = Y, Z = a\}$

Unifikation

- Gegeben: Menge von Termgleichungen
- Gesucht: Substitution, nach deren Anwendung alle Gleichungen erfüllt sind (Unifikator)
- $\{X = Y, Z = a\}$
  Lösungen z.B. $\sigma = [X \Rightarrow Y, Z \Rightarrow a]$, $\sigma = [Y \Rightarrow X, Z \Rightarrow a]$,
  $\sigma = [X \Rightarrow b, Y \Rightarrow b, Z \Rightarrow a]$

Unifikation

- Gegeben: Menge von Termgleichungen
- Gesucht: Substitution, nach deren Anwendung alle Gleichungen erfüllt sind (Unifikator)
- $\{X = Y, Z = a\}$
  Lösungen z.B. $\sigma = [X \Rightarrow Y, Z \Rightarrow a]$, $\sigma = [Y \Rightarrow X, Z \Rightarrow a]$,
  $\sigma = [X \Rightarrow b, Y \Rightarrow b, Z \Rightarrow a]$
- $\{X = f(Y), g(X, Y) = g(f(Z), b)\}$

Unifikation

- Gegeben: Menge von Termgleichungen
- Gesucht: Substitution, nach deren Anwendung alle Gleichungen erfüllt sind (Unifikator)
- $\{X = Y, Z = a\}$
  Lösungen z.B. $\sigma = [X \Rightarrow Y, Z \Rightarrow a]$, $\sigma = [Y \Rightarrow X, Z \Rightarrow a]$,
  $\sigma = [X \Rightarrow b, Y \Rightarrow b, Z \Rightarrow a]$
- $\{X = f(Y), g(X, Y) = g(f(Z), b)\}$
  Lösung $\sigma = [X \Rightarrow f(b), Y \Rightarrow b, Z \Rightarrow b]$

Unifikation

- Gegeben: Menge von Termgleichungen
- Gesucht: Substitution, nach deren Anwendung alle Gleichungen erfüllt sind (Unifikator)
- $\{X = Y, Z = a\}$
  Lösungen z.B. $\sigma = [X \Rightarrow Y, Z \Rightarrow a]$, $\sigma = [Y \Rightarrow X, Z \Rightarrow a]$,
  $\sigma = [X \Rightarrow b, Y \Rightarrow b, Z \Rightarrow a]$
- $\{X = f(Y), g(X, Y) = g(f(Z), b)\}$
  Lösung $\sigma = [X \Rightarrow f(b), Y \Rightarrow b, Z \Rightarrow b]$
- $\{X = f(Y), h(X) = h(g(a))\}$

Unifikation

- Gegeben: Menge von Termgleichungen
- Gesucht: Substitution, nach deren Anwendung alle Gleichungen erfüllt sind (Unifikator)
- $\{X = Y, Z = a\}$
  Lösungen z.B. $\sigma = [X \Rightarrow Y, Z \Rightarrow a]$, $\sigma = [Y \Rightarrow X, Z \Rightarrow a]$,
  $\sigma = [X \Rightarrow b, Y \Rightarrow b, Z \Rightarrow a]$
- $\{X = f(Y), g(X, Y) = g(f(Z), b)\}$
  Lösung $\sigma = [X \Rightarrow f(b), Y \Rightarrow b, Z \Rightarrow b]$
- $\{X = f(Y), h(X) = h(g(a))\}$
  Nicht unifizierbar! Widerspruch durch Gleichung $f(Y) = g(a)$!

Genaue Abarbeitung

```
mother(marge, lisa). mother(marge, bart).
father(homer, bart). father(grampa, homer).

parent(X, Y) :- father(X, Y).
parent(X, Y) :- mother(X, Y).

grandparent(X, Y) :- parent(X, Z), parent(Z, Y).
```

- Versuche, passende Regel zu finden
- Unifiziere den gesuchten Term mit Regelkopf
- Versuche, Teilziele von links nach rechts zu zeigen
- Bei Fehlschlag: versuche, Teilziel links davon auf andere Weise zu erfüllen
- Bei Fehlschlag für Regel: suche nächste Regel

Beispielabfrage `? grandparent(X, Y).`

Programme

- Prolog hat nur Prädikate und Erfüllbarkeit
- Insbesondere keine Funktionen mit Rückgabewerten
- Idee: mache Rückgabewert zum Teil des Prädikats

```
gegenueber(links, rechts).
gegenueber(rechts, links).

mirror(leaf, leaf).
mirror(node(X, Y), node(Y1, X1)) :-
    mirror(X, X1), mirror(Y, Y1).

symmetric(X) :- mirror(X, X).
```

Zusatz: Unifikationsziele

- Ziele der Form *Term* = *Term*
- Unifiziert die beiden Terme
- erfüllbar, falls Unifikation gelingt

Zusatz: Arithmetik

- Erinnerung: Terme stehen nur für sich selbst
- Daher Gleichungen wie $4 = 1 + 3$ nicht erfüllbar
- Daher: Spezielles Rechenprädikat *Term* **is** *Term*
- Rechte Seite darf nur Arithmetik und keine Variablen enthalten
- Rechte Seite wird ausgerechnet und mit der linken Seite unifiziert

Zusatz: Arithmetik

- Erinnerung: Terme stehen nur für sich selbst
- Daher Gleichungen wie $4 = 1 + 3$ nicht erfüllbar
- Daher: Spezielles Rechenprädikat *Term* **is** *Term*
- Rechte Seite darf nur Arithmetik und keine Variablen enthalten
- Rechte Seite wird ausgerechnet und mit der linken Seite unifiziert

Beispiele:

- ? X is 0. $\Longrightarrow$ X = 0.
- ? X is 1 + 3. $\Longrightarrow$ X = 4.
- ? 4 is 1 + 3. $\Longrightarrow$ yes.
- ? 5 is 1 + 3. $\Longrightarrow$ no.
- ? 1 + 3 is 1 + 3. $\Longrightarrow$ no.
- ? X is 1 + Y. $\Longrightarrow$ Exception
- ? X is homer. $\Longrightarrow$ Exception

Zusatz: Cut

- Spezialziel **!** (Ausrufezeichen)
- Gelingt zunächst
- Bei Reerfüllung: Fehlschlag der kompletten Regel
- inklusiver aller alternativen Versionen des Regelkopfes

```
a(X, Z) :- b(X, X, Z).
a(1, 3).

b(X, 1, Z) :- e(X, Z), !, f(Z).
b(X, 2, Z) :- e(X, Z), !, f(Z).
b(X, Y, 6).

e(1, 4).
e(1, 5).
e(2, 5).

f(5).
```