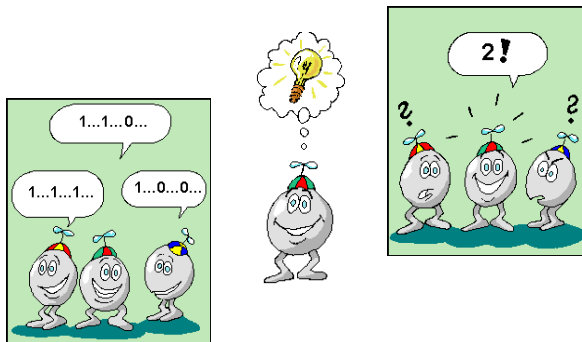


Einführung in Prolog



Arithmetik

- ohne die Operatordefinitionen müsste man z.B. $a :- b, c, d.$ als $:-(a, ,(b, ,(c,d)))$. schreiben
- mit Klammern lassen sich Vorrang und Assoziativität wie gewohnt beeinflussen
- in einem Ausdruck wird der Operator mit der schwächsten Bindung zum Hauptfunktork

Ralph Kötter - Einführung in Prolog, WS 2002/03

Vorlesung 6 - 1

Arithmetik

- arithmetische Ausdrücke werden nicht automatisch ausgewertet:
`?- X = 123+45.`

`X = 123+45`

`Yes`
- zur Auswertung braucht es ein spezielles Prädikat: `is`
`?- X is 123+45.`

`X = 168`

`Yes`

Ralph Kötter - Einführung in Prolog, WS 2002/03

Vorlesung 6 - 2

eigene Arithmetikprädikate

- mit Hilfe des Prädikats `arithmetic_function/1` können eigene Prädikate arithmetisch nutzbar gemacht werden
- die Prädikate benötigen dazu ein zusätzliches Argument für die Ergebnisübergabe:

```
% minimum(A,B, Ergebnis)
minimum(A,B, A) :- A < B.
minimum(A,B, B) :- A >= B.
?- arithmetic_function(minimum/2).
Yes
?- X is minimum(27,15).
X = 15
Yes
```

Ralph Kötter - Einführung in Prolog, WS 2002/03

Vorlesung 6 - 3

Die Beweisprozedur von Prolog

- der Beweiser in Prolog geht mechanisch vor
- die Suche nach einer passenden Klausel in der Wissensbasis erfolgt von oben nach unten
- die Reihenfolge der Klauseln in der Quelldatei spielt eine wichtige Rolle beim Programmieren
- die Quelldatei wird von oben nach unten durchsucht
- die oberste passende Klausel wird gewählt

Ralph Kötter - Einführung in Prolog, WS 2002/03

Vorlesung 6 - 4

Die Beweisprozedur von Prolog

- die entsprechende Stelle der Wissensbasis wird markiert
- es wird ein *Choicepoint* eingerichtet
- dieser enthält die Belegungen der Platzhalter
- Platzhalter nehmen in Teilbeweisen einen Wert an und bleiben dann unverändert
- für alternative Lösungen müssen diese Belegungen zurückgenommen werden

Ralph Kötter - Einführung in Prolog, WS 2002/03

Vorlesung 6 - 5

Die Beweisprozedur von Prolog

- das Verfahren, in einem Beweis Schritte und Instantiierungen zurückzunehmen, wird **Backtracking** genannt
- mit solchen Rückwärtsschritten kann sich das Beweissystem in einen Zustand mit anderen Platzhalterbelegungen bewegen, ohne dass die Bedingung verletzt ist, dass Platzhalter nur eine einmalige Wertzuweisung akzeptieren

Ralph Kötter - Einführung in Prolog, WS 2002/03

Vorlesung 6 - 6

Die Beweisprozedur von Prolog

- wird ein alternativer Beweis für ein Goal notwendig, so probiert das System am letzten angelegten Choicepoint die nächste passende Alternative und benutzt dabei die ursprünglichen Belegungen der Platzhalter
- gibt der aktuelle Choicepoint keine Alternativen mehr her, so wird erneutes Backtracking initiiert, das zum nächsthöheren Choicepoint führt

Ralph Kötter - Einführung in Prolog, WS 2002/03

Vorlesung 6 - 7

Die Beweisprozedur von Prolog

- die Abarbeitung von Prologprogrammen lässt sich grafisch in Form sogenannter Und-/Oder-Bäume darstellen
- Aufbau:
 - ◆ die Wurzel ist die zu beweisende Aussage
 - ◆ dessen Nachfolger sind die Klauselköpfe des passenden Prädikats
 - ◆ deren Nachfolger sind die Unteranfragen der Klauselrümpfe
 - ◆ rekursive Wiederholung für jede Unteranfrage

Ralph Kötter - Einführung in Prolog, WS 2002/03

Vorlesung 6 - 8

Die Beweisprozedur von Prolog

- Warum Und-/Oder-Baum?
 - ◆ die passenden Klauselköpfe des Prädikats stellen Alternativen dar (ODER-verknüpft)
 - ◆ die Klauselrümpfe müssen bekanntlich alle erfüllt sein (UND-verknüpft)
- die ODER-Knoten stellen die *Choicepoints* dar
- die Blätter des Baums bestehen aus Fakten

Ralph Kötter - Einführung in Prolog, WS 2002/03

Vorlesung 6 - 9

Die Beweisprozedur von Prolog

- Beispielbaum:
 - frau(maria). % maria ist eine frau
 - frau(michaela). % ...
 - frau(susanne).
 - frau(anke).
 - frau(erika).
 - mann(jens). % jens ist ein mann
 - mann(bernd). % ...
 - mutter(anke, maria). % anke ist die mutter von maria
 - mutter(maria, jens). %
 - mutter(erika, bernd).
 - mutter(maria, susanne).
 - vater(bernd, jens). % bernd ist der vater von jens
 - vater(bernd, susanne). %

Ralph Kötter - Einführung in Prolog, WS 2002/03

Vorlesung 6 - 10

Die Beweisprozedur von Prolog

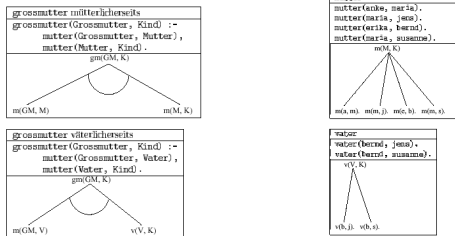
- Beispielbaum (Fortsetzung):
 - % mütterlicherseits:
 - % Grossmutter ist die grossmutter von Kind, wenn Grossmutter die % mutter von Mutter und Mutter die mutter von Kind ist.
 - %
 - % väterlicherseits:
 - % Grossmutter ist die grossmutter von Kind, wenn Grossmutter die % mutter von Vater und Vater der vater von Kind ist.
 - grossmutter(Grossmutter, Kind) :- % mütterlicherseits
 - mutter(Grossmutter, Mutter), mutter(Mutter, Kind).
 - grossmutter(Grossmutter, Kind) :- % väterlicherseits
 - mutter(Grossmutter, Vater), vater(Vater, Kind).

Ralph Kötter - Einführung in Prolog, WS 2002/03

Vorlesung 6 - 11

Die Beweisprozedur von Prolog

- Teilbäume:

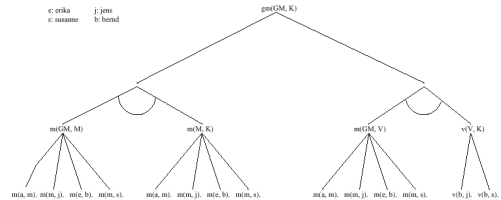


Ralph Kötter - Einführung in Prolog, WS 2002/03

Vorlesung 6 - 12

Die Beweisprozedur von Prolog

- komplett:



Ralph Kötter - Einführung in Prolog, WS 2002/03

Vorlesung 6 - 13

Die Beweisprozedur von Prolog

- auf der Suche nach einem Beweis werden die Knoten des Und-/Oder-Baumes in systematischer Weise besucht:
 - es wird an der Wurzel begonnen
 - das System wählt aus mehreren Alternativen die jeweils am weitesten links stehenden aus, die im bisherigen Teilbeweis noch nicht besucht worden ist
 - Und-verknüpfte Klauseln werden von links nach rechts abgearbeitet

Ralph Kötter - Einführung in Prolog, WS 2002/03

Vorlesung 6 - 14

Die Beweisprozedur von Prolog

- Anmerkungen:**
 - stellt man die Abbruchbedingung eines Prädikates hinter eine rekursive Klausel, die immer passt, wird der linke Ast des Beweisbaumes unendlich lang
 - die Abbruchbedingung ist in einem solchen Fall unerreichbar, der Beweis findet kein Ende
 - das System ist in einem endlosen Beweis verfangen
 - gerade bei rekursiven Prädikaten spielt daher die Klauselreihenfolge eine entscheidende Rolle, generell sollte man die Abbruchbedingung als erste Klausel notieren

Ralph Kötter - Einführung in Prolog, WS 2002/03

Vorlesung 6 - 15

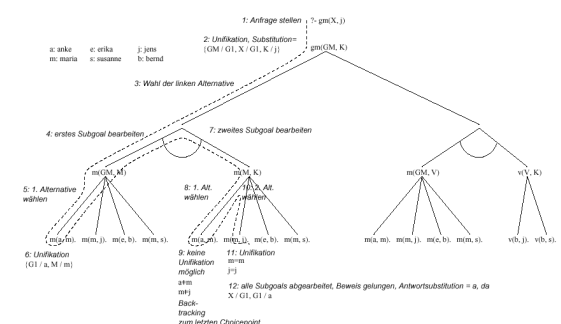
Die Beweisprozedur von Prolog

- Beispielbeweis:
- `grosstmutter(X,jens).`

Ralph Kötter - Einführung in Prolog, WS 2002/03

Vorlesung 6 - 16

Die Beweisprozedur von Prolog

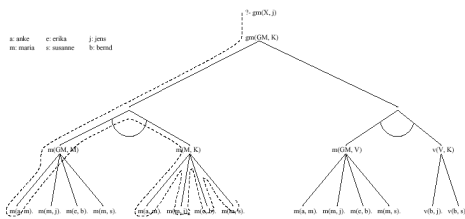


Ralph Kötter - Einführung in Prolog, WS 2002/03

Vorlesung 6 - 17

Die Beweisprozedur von Prolog

- Suche nach weiteren Lösungen

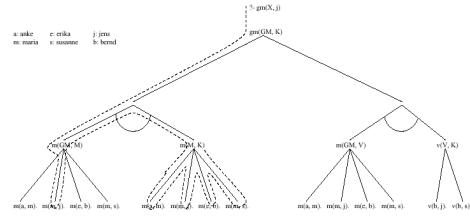


Ralph Kötter - Einführung in Prolog, WS 2002/03

Vorlesung 6 - 18

Die Beweisprozedur von Prolog

- Wahl einer neuen Alternative für das 1.Subgoal:



usw...

Ralph Kötter - Einführung in Prolog, WS 2002/03

Vorlesung 6 - 19

Die Beweisprozedur von Prolog

- Übung:
- Tracen des Aufrufs, Vergleichen der Ausgabe mit den Grafiken

```
?- spy(grossmutter).
% Spy point on grossmutter/2
```

Yes

```
[debug] ?- grossmutter(X,jens).
Call: (6) grossmutter(_G383, jens) ? creep
Call: (7) mutter(_G383, _G438) ? creep
Exit: (7) mutter(anke, maria) ? creep
....
```

Ralph Kötter - Einführung in Prolog, WS 2002/03

Vorlesung 6 - 20

Die Beweisprozedur von Prolog

- das Vierportmodell:
 - CALL (Teil-)Beweis wird angestoßen
 - EXIT (Teil-)Beweis erfolgreich beendet
 - FAIL (Teil-)Beweis ist gescheitert
 - REDO Wiederholung eines (Teil-)Beweises, dies führt im Allgemeinen zu einer alternativen Variablenbelegung

Ralph Kötter - Einführung in Prolog, WS 2002/03

Vorlesung 6 - 21

Die Beweisprozedur von Prolog

- das Prädikat **fail**
 - explizites Backtracking auslösen
 - „Dieser Beweis ist gescheitert, versuche eine Alternative!,,

- Programmierung einer Schleife mit **fail**:

```
?- member(_X, [1,2,3,4,5]), write(_X), nl, fail.
```

```
1
2
3
4
5
No
```

Ralph Kötter - Einführung in Prolog, WS 2002/03

Vorlesung 6 - 22

Die Beweisprozedur von Prolog

- Schleife mit Abbruchbedingung:

```
wiederhole(_).
bis(X,X).
bis(X,Y) :- not(X=Y), fail.
```

```
?- wiederhole(_), nat_zahl(_X), write(_X), nl, bis(_X, s(s(s(s(0))))).
0
s(0)
s(s(0))
s(s(s(0)))
s(s(s(s(0))))
Yes
```

Ralph Kötter - Einführung in Prolog, WS 2002/03

Vorlesung 6 - 23

Die Beweisprozedur von Prolog

- Übung: Ausprobieren und Erklären

Ralph Kötter - Einführung in Prolog, WS 2002/03

Vorlesung 6 - 24

Die Beweisprozedur von Prolog

- Der **Cut**
- Abschneiden aller verbleibenden Alternativen für alle Goals vor dem Cut
- kein Backtracking vor dem Cut
- hinter dem Cut bleibt alles normal

Ralph Kötter - Einführung in Prolog, WS 2002/03

Vorlesung 6 - 25

Die Beweisprozedur von Prolog

- Beispiel:
 - a :- b, c, !, d, e.
 - a :- f.
- zwei alternative Klauseln für **a**
- Cut nach Subgoals **b** und **c**
- gelingen b und c, so wird der Cut überschritten
- scheitern d oder e, so scheitert der gesamte Beweis!
 - ♦ die zweite Klausel (mit f) wird nicht ausgeführt
- scheitern b oder c, so wird die zweite Klausel versucht

Ralph Kötter - Einführung in Prolog, WS 2002/03

Vorlesung 6 - 26

Die Beweisprozedur von Prolog

- Der **Cut**
- „guarded gate“-Metapher:
 - Im Klauselrumpf stehen vor dem Cut Bedingungen, die erfüllt sein müssen, um den Cut zu überschreiten.
 - Bedingungen = „Torwächter“
 - Cut = Tor
 - danach fällt das Tor ins Schloss, keine Rückkehr möglich
 - => kein Backtracking vor dem Cut

Ralph Kötter - Einführung in Prolog, WS 2002/03

Vorlesung 6 - 27

Die Beweisprozedur von Prolog

- weitere Anwendung: Ergebnisausgabe
 - typ(X) :- is_list(X), !, write('Liste'), nl.
 - typ(_) :- write('keine Liste...'), nl.
- gelingt is_list, dann Ausgabe (nach Cut)
- keine Backtracking, keine weitere Ausgabe
- scheitert is_list, dann Ausgabe ‚keine Liste‘

Ralph Kötter - Einführung in Prolog, WS 2002/03

Vorlesung 6 - 28

Die Beweisprozedur von Prolog

- „rote“ und „grüne“ Cuts
- schneidet man nur Teile eines Beweisbaumes ab, die keine Lösung liefern, ändert sich die Bedeutung (Semantik) des Programms nicht (**grüner Cut**) .
 - ♦ Einsatz zur Effizienzsteigerung von Prologprogrammen, indem sie den Beweiser von überflüssiger Arbeit abhalten
- Schneidet man Teile des Beweisbaumes ab, die zu möglichen Lösungen führen, so ändert man die Semantik des Programms (**roter Cut**)

Ralph Kötter - Einführung in Prolog, WS 2002/03

Vorlesung 6 - 29

Die Beweisprozedur von Prolog

- cut-fail Kombinationen zur Simulation von Negation
- vorstellbar sind Prädikate, die unter bestimmten Bedingungen sofort und ohne Alternativen scheitern sollen
- Beispiel:
Prädikat **schmeckt_kindern_gut(X)**
- Expertenwissen:
Rosenkohl hat keine Chance

Ralph Kötter - Einführung in Prolog, WS 2002/03

Vorlesung 6 - 30

Die Beweisprozedur von Prolog

```
schmeckt_kindern_gut(X) :- rosenkohl(X), !, fail.  
schmeckt_kindern_gut(X) :- suess(X), bunt(X),  
                           ungesund(X).
```

```
rosenkohl(rosenkohl).  
suess(gummibaerchen).  
bunt(gummibaerchen).  
ungesund(gummibaerchen).
```

```
?- schmeckt_kindern_gut(rosenkohl).  
No  
?- schmeckt_kindern_gut(gummibaerchen).  
Yes
```

Ralph Kötter - Einführung in Prolog, WS 2002/03

Vorlesung 6 - 31

Die Beweisprozedur von Prolog

- Übung:
- Umschreiben in eine Regel mit not/1. Also ohne cut/fail.
- Warum liefert
?- schmeckt_kindern_gut(X).

No

????

Ralph Kötter - Einführung in Prolog, WS 2002/03

Vorlesung 6 - 32

Die Beweisprozedur von Prolog

- Aspekte des Cut:
 - ◆ Der Cut kann Programme effizienter aber auch schwerer nachvollziehbar machen.
 - ◆ Von der Verwendung allzu vieler Cuts wird im Allgemeinen abgeraten.
 - ◆ Aus logischer Sicht sind Cuts natürlich unangenehm. Prozedurale und deklarative Interpretationen können sich wesentlich unterscheiden.
 - ◆ Cuts ermöglichen deterministische Prädikate, d.h. Prädikate, die nur eine Lösung liefern.
 - ◆ ! und fail ermöglichen Konstruktionen, die IF-THEN-ELSE aus anderen Sprachen entsprechen.

Ralph Kötter - Einführung in Prolog, WS 2002/03

Vorlesung 6 - 33