Beweisen mit Prolog

Übersicht

- Unifikation
- Substitution
  - Instanz und Variablenbindung
- Wie werden Anfragen bewiesen?
  - Beweisziele
  - Passende Klauseln
  - Beweisregel für Fakten
  - Beweisregel für Regeln
  - Backtracking
- Beweisbaum/Suchbaum

Das Problem

Wie kann Prolog aus Fakten und Regeln…

```
person(hans).
person(gabi).
person(klara).
```

```
weiblich(klara).
weiblich(gabi).
```

```
frau(X) :-
  person(X),
  weiblich(X).
```

…Anfragen beantworten?

Anders gesagt

Wie beweist Prolog, dass 'frau(klara)' und 'frau(gabi)' aus obiger Wissenbasis folgen?

```
?- frau(Wer).
Wer = klara ? ;
Wer = gabi ? ;
no
```

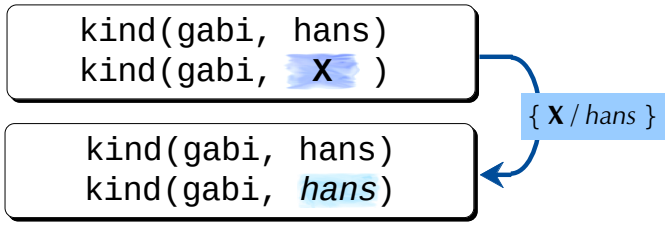

Ingredienzen

Was braucht Prolog zum Beweisen?

- **Termmanipulation**
  - Unifikation
  - Substitution
- **Beweisregeln**
  - für Fakten und für Regeln
- **Suchstrategie**
  - von oben nach unten
- **Backtracking**
  - Entscheidungspunkte

Termmanipulation: Unifikation

Unifikation versucht, zwei Terme gleich zu machen, indem Variablen so weit wie nötig ersetzt werden.



Bei Ersetzung (*Substitution*) von **X** durch *hans* werden die beiden Terme gleich.

Termmanipulation: Substitution

Im Term *T* eine Variable *V* durch Term *S* **substituieren**, heisst **alle** Vorkommen von *V* in *T* durch *S* ersetzen.

Schematisch

$$T' = T\{V/S\}$$

kind(gabi, hans) = kind(gabi,X) {X/hans}

Instanz

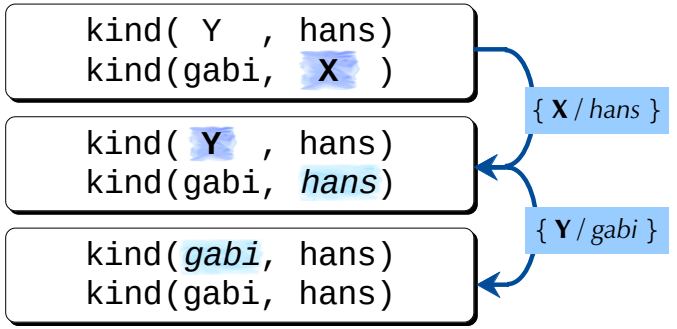
T' heisst **Instanz** von *T*. *T* wurde **instantiiert** zu *T'*.

Variablenbindung

Wenn beim Beweisen eine Variable *V* durch einen Term *S* ersetzt wird, spricht man davon, dass *V* an *S* **gebunden** wurde.

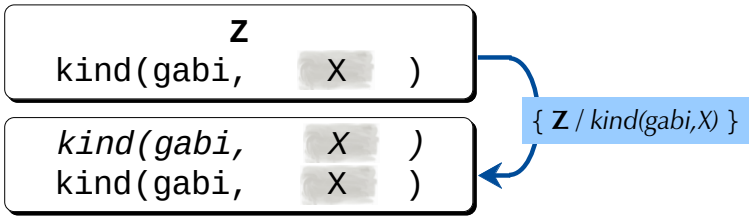
Unifikation durch Substitution

Manchmal müssen mehrere Variablen substituiert werden, um die beiden Terme identisch zu machen.



Unifikation durch Substitution

Es brauchen nicht *alle* Variablen substituiert zu werden.



- ***Aber** es müssen alle* Vorkommen *einer Variable ersetzt werden.*

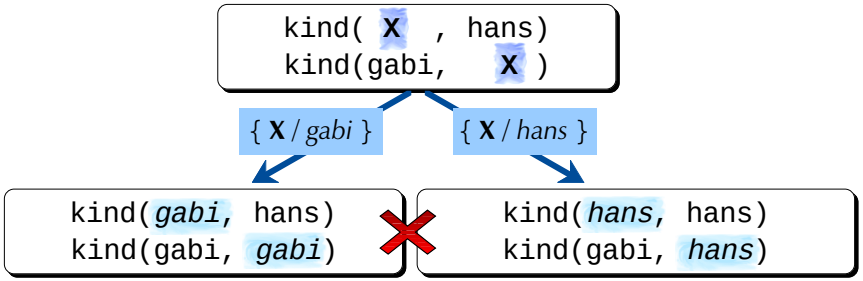
Nicht unifizierbare Terme

Manchmal gibt es keine Möglichkeit, Variablen zu ersetzen, damit zwei Terme identisch werden:

Die Unifikation **scheitert** *(unification failure)*.

Nicht unifizierbare Terme

Manchmal gibt es keine Möglichkeit, Variablen so zu ersetzen, damit zwei Terme identisch werden:



Unifizierbarkeit

Zwei Terme *T* und *U* sind unifizierbar, genau dann wenn gilt:

- *T* und *U* sind identische atomare Terme,
- **oder** *T* oder *U* ist eine Variable,
  - *substituiere alle Vorkommen der Variable*
- **oder** *T* und *U* sind komplexe Terme, wobei gilt:
  - *T* und *U* haben identische Hauptfunktoren,
  - **und** *T* und *U* haben dieselbe Stelligkeit,
  - **und** die einzelnen Argumente sind paarweise unifizierbar.

Unifizierbar oder nicht?

`p4711 = p4711` { }

Ja. (dasselbe Atom)

`X = fido` { X / *fido* }

Ja. (ein Variable)

`X = Y` { X / Y } { Y / X }

Ja. (2 Variablen: 2 Möglichkeiten)

`kind(gabi,X) = kind(Y,hans)` { X / *hans*, Y / *gabi*}

Ja. (gleicher Funktor kind/2 und paarweise unfizierbare Argumente)

Unifikation in Prolog

Das eingebaute zweistellige Prädikat = ist wahr, wenn seine Argumente unifizierbar sind.

```
?- hans = klara.
no
```

nicht unifizierbar

```
?- hans = hans.
yes
```

unifizierbar

```
?- kind(Y,hans) =
   kind(Y,hans).
true ?
yes
```

unifizierbar, keine Variablenbindung

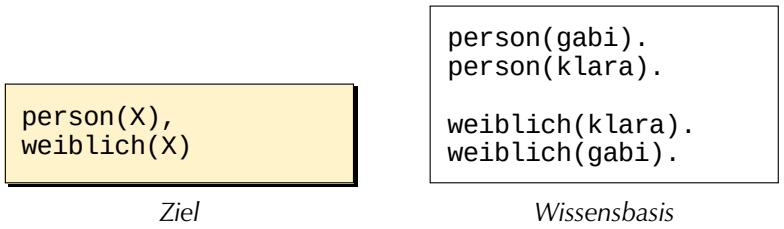
```
?- kind(gabi, X) =
   kind(Y, hans).
X = hans,
Y = gabi ?
yes
```

unifizierbar, mit Variablenbindung

Wie wird eine Anfrage bewiesen?

Nimm die Anfrage als Beweisziel (*goal*).

```
?- person(X), weiblich(X).
```

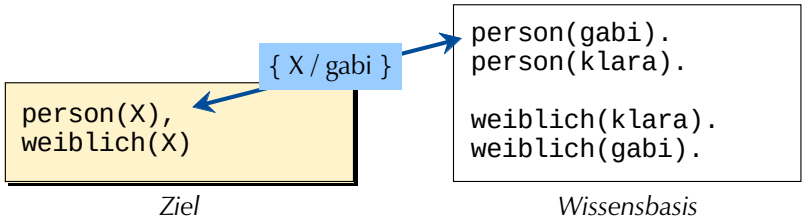


Passendes Fakt suchen

A. Suche ein Fakt, das mit dem ersten Term des Ziels unifiziert.

- Suchrichtung ist von oben nach unten!

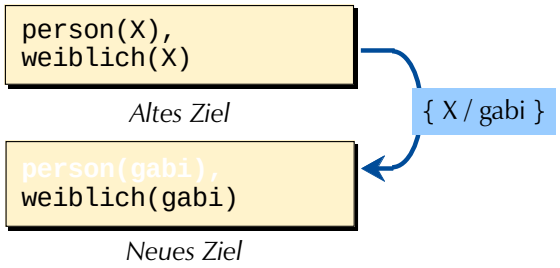
B. Merke dir die Substitution.



Beweisregel für Fakten

A. Mache die Substitution in allen Termen des Ziels.

B. Lösche den ersten Term des Ziels.

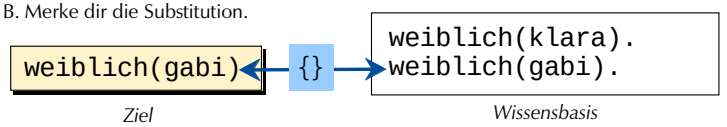


Da Capo Al Fine

Passende Klausel suchen

A. Suche ein Fakt, das mit dem ersten Term des Ziels unifiziert.

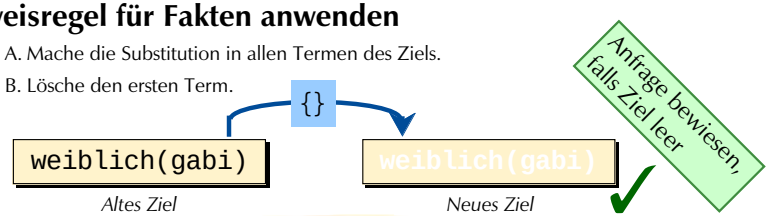
B. Merke dir die Substitution.



Beweisregel für Fakten anwenden

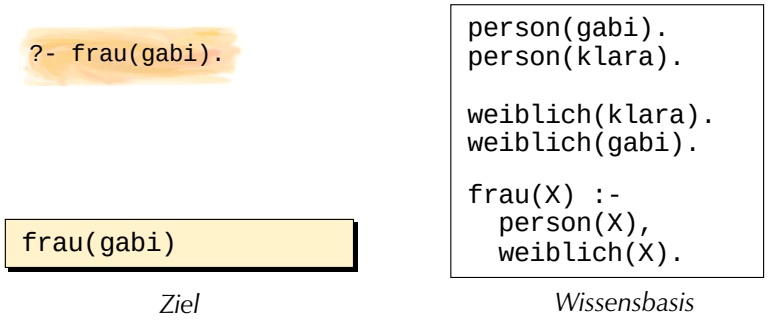
A. Mache die Substitution in allen Termen des Ziels.

B. Lösche den ersten Term.



Wie wird eine Anfrage bewiesen?

Nimm die Anfrage als Beweisziel.

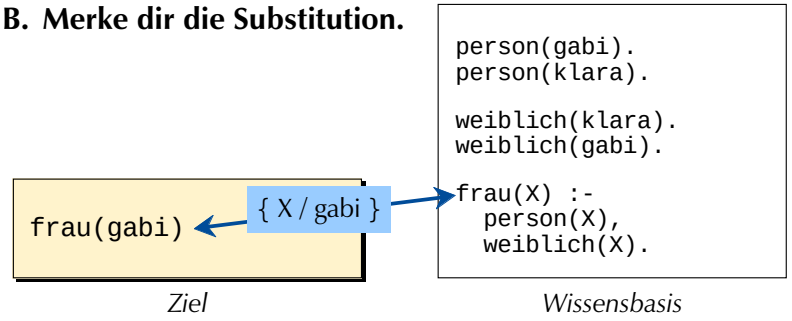


Passende Regel suchen

A. Suche einen Regelkopf, der mit dem ersten Term des Ziels unifiziert.

- Suchrichtung ist von oben nach unten!

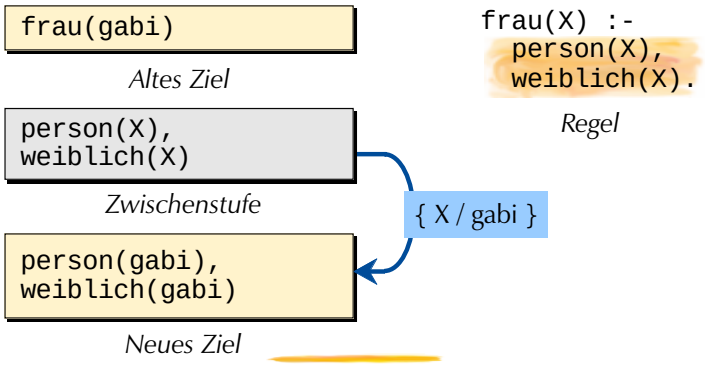
B. Merke dir die Substitution.



Beweisregel für Regeln

A. Ersetze den ersten Term des Ziels durch den Regelrumpf.

B. Mache die Substitution in allen Termen des Ziels.



Beweisen mit Fakten und Regeln

Beweisverfahren im Überblick

I. Suche von oben nach unten die erste passende Klausel für den ersten Term des Ziels.
   a. Falls Klausel ein Faktum ist, wende die Beweisregel für Fakten an.
      i. Wenn das Ziel leer ist, dann ist der Beweis gelungen. Ende
      ii. Wenn das Ziel nicht leer ist, beweise es gemäss I.
   b. Falls Klausel eine Regel ist, wende die Beweisregel für Regeln an.
      i. Beweise Ziel gemäss I.
   c. Falls keine passende Klausel mehr gefunden werden kann, gelingt der Beweis nicht.

- *ad c) Durch Backtracking werden alternative passende Klauseln aufgespürt.*

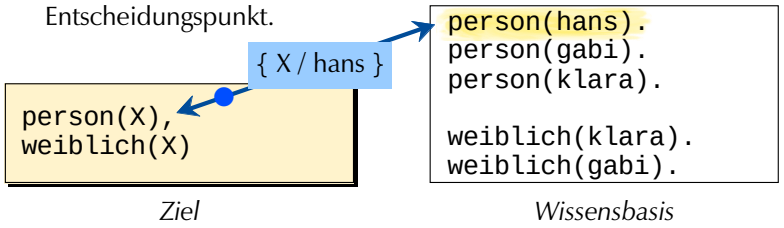
Wie wird eine Anfrage bewiesen?

Nimm die Anfrage als Beweisziel.

```
?- person(X), weiblich(X).
```

Suche passende Klausel.

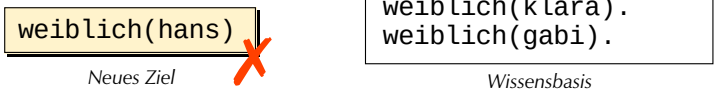
- Bei mehreren passenden Klauseln merkt sich Prolog einen Entscheidungspunkt.



Ziel *Wissensbasis*

Backtracking aus Sackgasse

Durch Anwenden der Beweisregel für Fakten entsteht ein neues Ziel.



Neues Ziel *Wissensbasis*

Aber: Beweis ist in der Sackgasse

- Keine passende Klausel für neues Ziel!

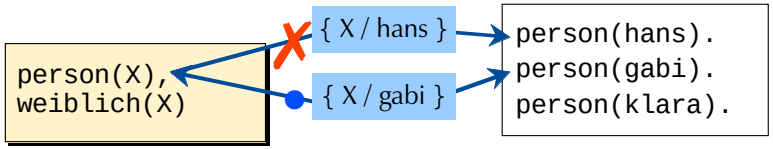
Deshalb: Backtracking (Rückverfolgen)

- Gehe zum zuletzt gemachten Entscheidungspunkt zurück!
- Mache dabei die unterwegs erfolgten Variablenbindungen rückgängig!

Backtracking

Beweise das Ziel, wo zuletzt ein Entscheidungspunkt (*decision point*) gesetzt wurde.

- Markiere bearbeiteten Entscheidungspunkt als erledigt.
- Setze einen neuen Entscheidungspunkt, falls immer noch mehrere Klauseln passen.
  - *Hinweis: Auch die Bindungen, die beim Setzen des alten Entscheidungspunkts entstanden sind, werden rückgängig gemacht!*



Ziel *Passende Klauseln*

Manuelles Backtracking

Die manuelle Eingabe des Strichpunkts am Prompt des Prolog-Interpreters löst Backtracking aus.

```
?- person(X), weiblich(X).
X = gabi ? ;
X = klara ? ;
no
```

- Die für jeden gelungenen Beweis erforderlichen Variablenbindungen werden vom Prolog-Interpreter jeweils herausgeschrieben.

Visualisierung als Beweis-/Suchbaum

Bestandteile und Legende

- Beweisziele
  - «Aktuelles Beweisziel: frau(Wer)»

frau(Wer)

- Beweis gefunden

✓

- Passende Prädikatsklausel mit Substitution
  - «Beweis für ersten Term des Ziels durch 2. Klausel des einstelligen Prädikats *person*, wobei die Variable *Wer* durch *gabi* ersetzt wurde.» *person/1/2 {Wer/gabi}*
  - *Substitutionen von Variablen, die beim Beweisen mit Regeln entstanden sind, aber in Zielen nie auftauchen, können weggelassen werden.*
- Sackgasse

✗

Visualisierung als Beweis-/Suchbaum