

Algorithmische Logik / Computational Logic

Ralf Möller, TUHH

- Beim vorigen Mal:
 - Prädikatenlogik: Syntax, Semantik, Entscheidungsprobleme
- Heute:
 - Prädikatenlogik: Algorithmus für Erfüllbarkeitsproblem
- Lernziele:
 - Beweisverfahren für Prädikatenlogik

Danksagung

- Bildmaterial: S. Russell, P. Norvig, Artificial Intelligence: A Modern Approach, Prentice Hall, 1995 (AIMA)
- Präsentationen nach AIMA Kap. 3 sind orientiert an Präsentationen von Jörg Desel, Lehrstuhl für Angewandte Informatik, Katholische Universität Eichstätt

Normalform

Konjunktive und *disjunktive* Normalform (KNF und DNF) sind wie in der Aussagenlogik definiert.

Die Herstellung der Normalformen erfordert zusätzlich die Elimination der Quantoren.

Pränexnormalform

Erster Schritt: Bilden der **Pränexnormalform** einer Formel

Quantorenpräfix + Matrix

$$\forall x_1 \forall x_2 \exists x_3 \dots \forall x_n (\varphi)$$

Erzeugen der Pränexnormalform durch Äquivalenzumformungen:

1. Elimination von \Leftrightarrow und \Rightarrow
2. \neg vor die Atome
3. Quantoren nach außen

Beispiel

$$\neg \forall x [(\forall x P(x)) \Rightarrow Q(x)]$$
$$\neg \forall x [\neg(\forall x P(x)) \vee Q(x)]$$
$$\exists x [(\forall x P(x)) \wedge \neg Q(x)]$$

Variablennormalisierung

Variablenumbenennung

Lemma: Sei y eine Variable, die nicht in φ vorkommt.

Dann gilt $\forall x \varphi \equiv \forall y \varphi [x/y]$ und $\exists x \varphi \equiv \exists y \varphi [x/y]$.

Variablennormalisierung:

Je zwei Variablen hinter Quantoren
werden durch Variablenumbenennung verschieden
gemacht.

Skolemisierung (1)

Elimination von Existenzquantoren

Eine Formel der Form

$$\forall x_1 \dots x_n \exists y \varphi$$

Wird ersetzt durch die Formel

$$\forall x_1 \dots x_n \varphi',$$

Wobei φ' aus φ dadurch entsteht, dass in φ jedes freie Vorkommen von y durch $f(x_1 \dots x_n)$ ersetzt wird.

f ist ein „neues“ n -stelliges Funktionssymbol (**Skolemfunktion**)

Der **Skolemterm** $f(x_1 \dots x_n)$ repräsentiert ein y , das zu den gegebenen $x_1 \dots x_n$ existieren soll, d.h. ein **Beispiel**.

Skolemisierung (2)

Beispiel: $\forall x \exists y [P(x) \Rightarrow Q(y)]$

Skolemisierung: $\forall x [P(x) \Rightarrow Q(f(x))]$

Skolemnormalform: Skolemisierte Pränexnormalform

Beispiel: $\exists x ((\forall x P(x)) \wedge \neg Q(x))$

$\exists y ((\forall x P(x)) \wedge \neg Q(y))$

$\exists y (\forall x (P(x) \wedge \neg Q(y)))$

$\forall x (P(x) \wedge \neg Q(f))$

Satz: Zu jeder geschlossenen Formel φ
kann ihre Skolemnormalform effektiv berechnet werden.

Skolemisierung (3)

Die Skolemisierung ist keine äquivalente Umformung, d.h. sie ist nicht modellerhaltend.

Satz: Eine Formel ist genau dann erfüllbar bzw. unerfüllbar, wenn ihre Skolemnormalform erfüllbar bzw. unerfüllbar ist.

Wenn wir mit einem negativen (Test-) Kalkül arbeiten, also den Nachweis der Unerfüllbarkeit von Formeln anstreben, können statt allgemeiner prädikatenlogischer Formeln deren Skolemnormalformen verwendet werden.

Erzeugen der Klauselformel

Eine prädikatenlogische Formel wird durch folgende Transformationsschritte in **Klauselform** gebracht:

- \Leftrightarrow und \Rightarrow eliminieren
- \neg vor die Atome bringen
- gebundene Variablen umbenennen
- Herstellen der Pränexnormalform
- Skolemisierung
- Matrix in konjunktive Normalform bringen
- Allquantoren entfernen
- Überführen in Mengenschreibweise

Beispiel

$$[\exists x \neg P(x) \vee \forall x Q(x)] \Rightarrow \forall x R(x)$$

\Rightarrow eliminieren

$$\neg[\exists x \neg P(x) \vee \forall x Q(x)] \vee \forall x R(x)$$

\neg vor die Atome

$$[\neg\exists x \neg P(x) \wedge \neg\forall x Q(x)] \vee \forall x R(x)$$

$$[\forall x P(x) \wedge \exists x \neg Q(x)] \vee \forall x R(x)$$

Variablen umbenennen

$$[\forall x P(x) \wedge \exists y \neg Q(y)] \vee \forall z R(z)$$

Pränexnormalform

Beispiel (2)

Pränexnormalform

$$\forall x \exists y \forall z [(P(x) \wedge \neg Q(y)) \vee R(z)]$$

Skolemisierung

$$\forall x, z [(P(x) \wedge \neg Q(f(x))) \vee R(z)]$$

Matrix in KNF

$$\forall x, z [(P(x) \vee R(z)) \wedge (\neg Q(f(x)) \vee R(z))]$$

Quantoren eliminieren, Klauselform

$$\{\{\{P(x), R(z)\}, \{\neg Q(f(x)), R(z)\}\}$$

Resolution (1)

Voraussetzung: Klauselform

Beispiel:

$$\{\neg P(x, f(y))\} \quad \{P(z, f(g(z)))\}$$

Wann ergibt sich ein Widerspruch?

Aus dieser Klauselmengemenge kann nur dann die leere Klausel abgeleitet werden, wenn die beiden Atome $P(x, f(y))$ und $P(z, f(g(z)))$ identifiziert werden.

Substitution: Ersetzung der Variablen durch Terme, so dass beide Atome gleich werden.

Beispiel: ersetze y durch $g(z)$
ersetze x durch z

Substitution

$\sigma = [x_1/t_1, x_2/t_2, \dots, x_n/t_n]$ ist eine Substitution

(ersetze x_1 durch t_1 , x_2 durch t_2 , ..., x_n durch t_n).

Für eine atomare Formel α ist

$\sigma\alpha$ die Anwendung der Substitution σ auf α ,
die alle Vorkommen der Variablen x_i in α durch die
entsprechenden t_i ersetzt.

Beispiel:

$$\varphi = P(f(x), y)$$

$$\sigma = [x/z, y/f(z)]$$

$$\sigma\varphi = P(f(z), f(z))$$

Unifikation (1)

Unifikator zweier atomarer Formeln α_1 und α_2 :

Substitution σ , die α_1 und α_2 **unifiziert** (gleich macht),
d.h. es soll gelten: $\sigma\alpha_1 = \sigma\alpha_2$.

Allgemeinster Unifikator σ (engl. *most general unifier, mgu*):

alle anderen Unifikatoren ergeben sich durch
Instantiierung der Variablen in σ , d.h.:

Zu jedem Unifikator σ' gibt es eine Substitution τ mit

$$\sigma' \alpha_1 = \tau(\sigma\alpha_1) = \tau(\sigma\alpha_2) = \sigma' \alpha_2.$$

Satz:

Für je zwei Ausdrücke gibt es, bis auf Variablenumbenennung,
höchstens einen allgemeinsten Unifikator.

Unifikation (2)

Beispiel:

Unifikation der Atome $Q(f(x), v, b)$ und $Q(f(a), g(u), y)$
durch die beiden Substitutionen

$$\sigma = [x/a, v/g(u), y/b]$$

$$\sigma' = [x/a, v/g(c), u/c, y/b]$$

σ ist allgemeinsten Unifaktor.

Beachte:

Die Terme x und $f(x)$ sind **nicht** unifizierbar (**occur-check**)

Die Terme $f(x)$ und $g(x)$ sind nicht unifizierbar.

Unifikation (3)

Algorithmus zur Berechnung des mgu für
eine Menge von Atomen $\{\alpha_i\}$.

$\sigma = []$.

LOOP IF alle $\sigma\alpha_i$ identisch **THEN**

RETURN σ (ein mgu)

Bilde Unterscheidungs Menge (engl. Disagreement Set, DS),

Bsp.: $DS(\{P(c, f(c, g(z), \dots)) \dots P(c, f(c, u, \dots))\} = \{g(z), u\}$

Wähle Variable $x \in DS$ und Term $t \in DS$, der x nicht enthält

IF dies nicht möglich ist **THEN**

RETURN failure ($\{\alpha_i\}$ nicht unifizierbar)

Ergänze σ um $[x/t]$

ENDLOOP

Beispiele

Sind folgende Atome unifizierbar?

$P(x)$ und $Q(y)$

$P(x, y)$ und $P(z)$

$P(x, y)$ und $P(a, f(a))$

$P(x, y)$ und $P(f(z), g(z))$

$P(x, f(y))$ und $P(z, f(g(z)))$

$P(x, x)$ und $P(f(y), f(g(z)))$

$P(x, f(x))$ und $P(y, y)$

$P(x, a)$ und $P(b, x)$

$P(x, f(x, x), z, f(z, z))$ und $P(f(a, a), y, f(y, y), u)$

Resolution (2)

$$\frac{C_1 \cup \{l\}, C_2 \cup \{\bar{l}\}}{\sigma(C_1 \cup C_2)}$$

$C_1 \cup \{l\}$ und $C_2 \cup \{\bar{l}\}$ sind variablendisjunkt und σ ist ein allgemeinsten Unifikator für l und \bar{l} , d.h. $\sigma l = \sigma \bar{l}$

Beispiel: $\{\{P(x), P(f(y)), Q(g(x))\}, \{R(f(z)), \neg P(f(z))\}\},$

$$\sigma = [x/f(v), y/v, z/v]$$

$$\{Q(g(f(v))), R(f(v))\}$$

Satz:

Eine prädikatenlogische Klauselmengemenge Δ ist unerfüllbar gdw. im prädikatenlogischen Resolutionskalkül die leere Klausel aus Δ abgeleitet werden kann, $\Delta \mid \sim$.

Beispiel 1

WB:

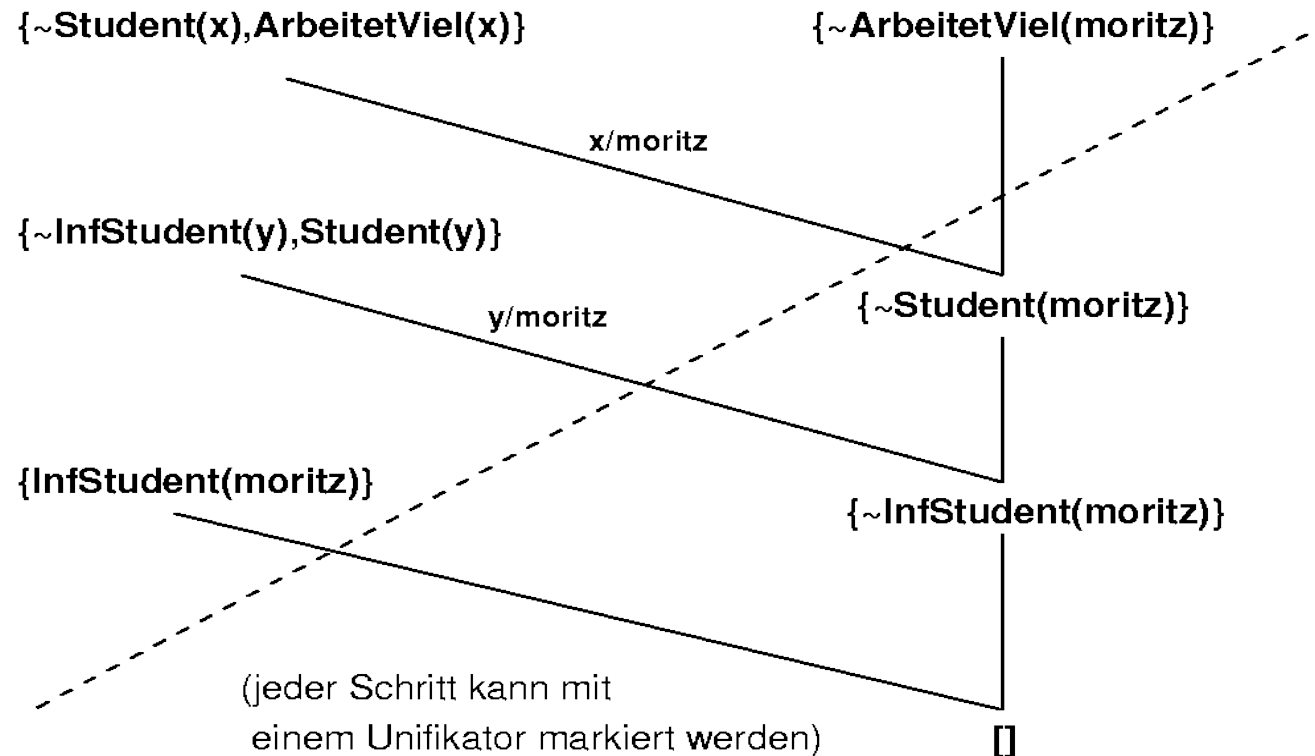
$$\forall x (InfStudent(x) \Rightarrow Student(x))$$

$$\forall x (Student(x) \Rightarrow ArbeitetViel(x))$$

$$InfStudent(moritz)$$

Frage:

$$ArbeitetViel(moritz)$$



Beispiel 2

WB:

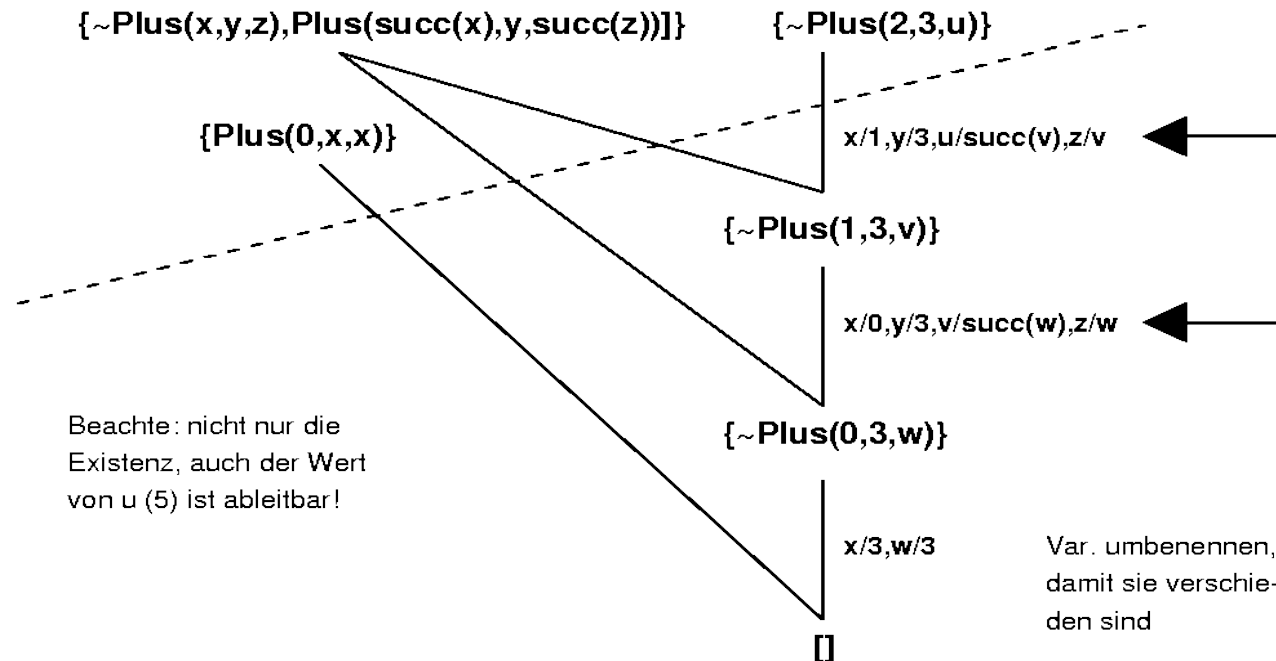
$$\forall x [Plus(null, x, x)]$$

$$\forall x, y, z [Plus(x, y, z) \Rightarrow Plus(succ(x), y, succ(z))]$$

Frage:

$$\exists u Plus(2, 3, u)$$

wobei wir der besseren Lesbarkeit wegen schreiben:
 0 statt *null*, 2 statt *succ(succ(null))* usw.



Faktorisierung (2)

Faktorisierung: Verfeinerung der Resolutionsregel

Verschmelzung der Literale in den Elternklauseln durch Anwendung einer geeigneten Substitution, bevor die Resolvente erzeugt wird.

Faktorisierungsregel:

$$\square \square = [x/\textit{barbier}] \quad \frac{\{\square \textit{Rasiert}(\textit{barbier}, x), \square \textit{Rasiert}(x, x)\}}{F1 : \{\square \textit{Rasiert}(\textit{barbier}, \textit{barbier})\}}$$

$$\square \square \square = [y/\textit{barbier}] \quad \frac{\{\textit{Rasiert}(y, y), \textit{Rasiert}(\textit{barbier}, y)\}}{F2 : \{\textit{Rasiert}(\textit{barbier}, \textit{barbier})\}}$$

Aus $F1$ und $F2$ kann die leere Klausel abgeleitet werden.

Entscheidungsproblem Folgerbarkeit

- Sei eine Formelmenge H , eine Menge von Hypothesen, gegeben
- Eine Formel F ist eine Folgerung aus H , geschrieben $H \models F$, genau dann, wenn alle Modelle der Formeln in H auch Modelle von F sind.

Begriffe

semantisch

Folgerung

\models

Erfüllbarkeit

Unerfüllbarkeit

syntaktisch

Ableitbarkeit

\vdash

Konsistenz

Inkonsistenz

Beziehungen zwischen Entscheidungsproblemen

- Eine Formel F ist eine Folgerung aus einer Formelmenge H , genau dann wenn $H \cup \{\neg F\}$ unerfüllbar ist.
- Resolution kann also auch das Folgerbarkeitsproblem entscheiden
- Formelmenge H heißt auch Wissensbasis

Generierung von Antworten (ask)

Gegeben sei ein Resolutionsbeweis für $\exists x P(x)$,
d.h. für gewisse Werte von x ist die Formel unerfüllbar.

Wie sieht „*Beispiel*“ für x aus, so dass Formel unerfüllbar?

Statt die verwendeten Unifikatoren durchzusuchen,
können **Antwortprädikate** verwendet werden.

Sie akkumulieren die während des Beweises konstruierte
Substitution für x .

Das Antwortprädikat A kommt in der Signatur noch nicht vor.

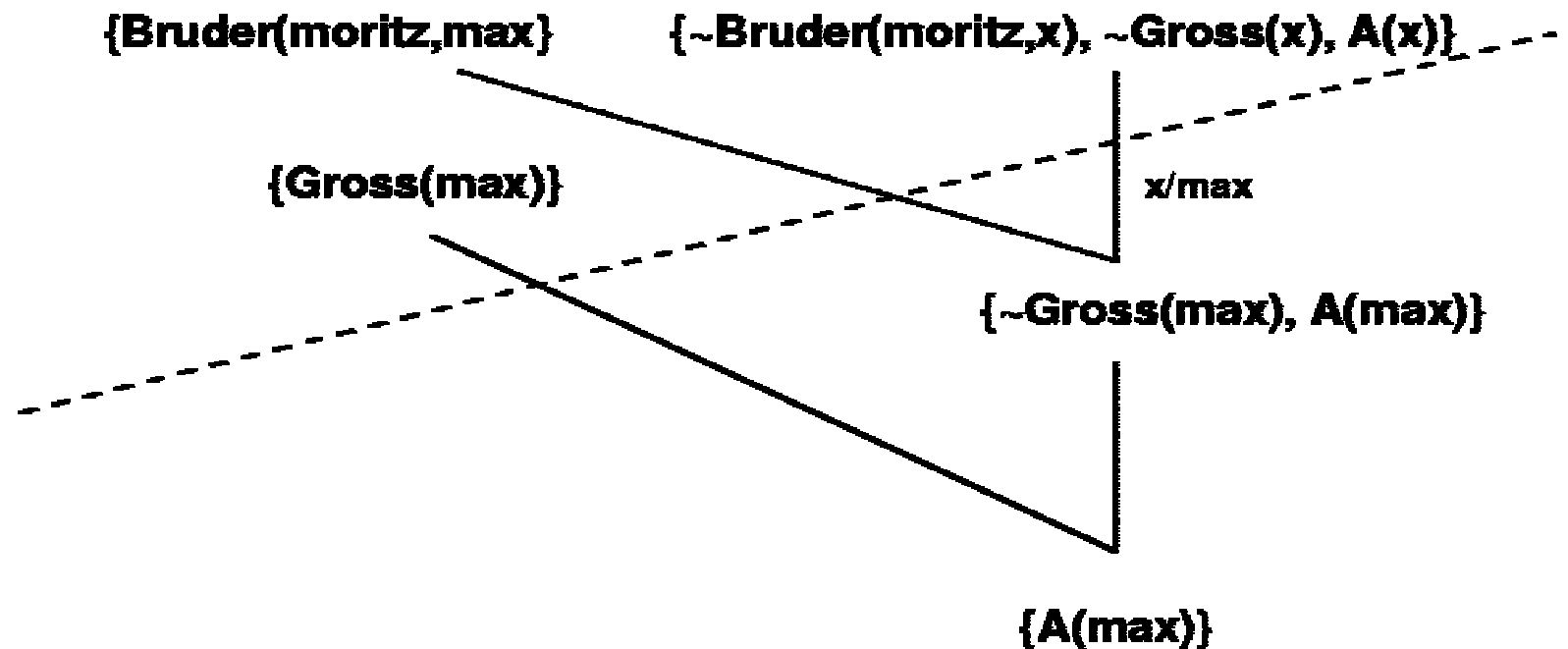
- Ersetze die Formel $\exists x P(x)$ durch $\exists x [P(x) \wedge \neg A(x)]$.
- Der Resolutionsbeweis erzeugt statt $_$ eine Klausel, die nur das Antwortprädikat enthält.

Beispiel 3 WB:

Bruder(moritz, max), Bruder(moritz, fritz) ...
Gross(max), Gross(fritz) ...

Frage:

$\exists x [Bruder(moritz, x) \wedge Gross(x)]$



Behandlung von = bei der Resolution

- Über Unifikation ...

Funktionen als Datenstrukturen (1)

- $\{\}$ = Emptyset
- $\{x\}$ = Adjoin(x , Emptyset)
- $\{x, y\}$ = Adjoin(x , Adjoin(y , Emptyset))
- $\{x, y \mid s\}$ = Adjoin(x , Adjoin(y , s))
- $r \cup s$ = Union(r , s)
- $r \cap s$ = Intersection(r , s)

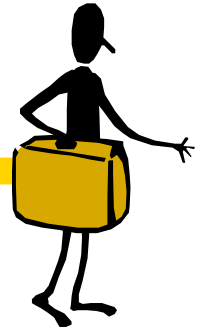
Funktionen als Datenstrukturen (2)

- `[]` = `Nil`
- `[x]` = `Cons(x, Nil)`
- `[x, y]` = `Cons(x, Cons(y, Nil))`
- `[x, y | list]` = `Cons(x, Cons(y, list))`

Spezielle Prädikate

- $x \in s$ = Member(x, s)
- $r \sqsubseteq s$ = Subset(r, s)

Zusammenfassung



- Algorithmus für Erfüllbarkeit: Resolution