
10. Resolution und Prolog

Prof. Dr. Rudolf Kruse

University of Magdeburg

Faculty of Computer Science

Magdeburg, Germany

rudolf.kruse@cs.uni-magdeburg.de



Resolventenprinzip und Prolog

- **Betrachtung formaler Ansätze zur Wissensmodellierung und Problemlösung**
- **Einsatzgebiete z.B. automatische Beweiser, Beschreibungen in der Robotik**
- **Im Folgenden u.a.:**
 - **Aussagenkalkül**
 - ◆ **Resolutionsregel / Resolventenverfahren**
 - ◆ **Horn-Klauseln**
 - **Prädikatenkalkül**
 - ◆ **Unifikation**
 - ◆ **Resolution**

Aussagenkalkül (propositional calculus)

10.1 Begriffsbildung

- Atome: **Bat_OK, Lifiable, Moves, ...**
- Literale sind entweder Atome (positive Literale) oder negierte Atome (negative Literale): **Bat_OK, $\neg$ Bat_OK, Lifiable, $\neg$ Lifiable, Moves, $\neg$ Moves, ...**
- Klauseln sind Disjunktionen von Literalen (als Abkürzung von $L_1 \vee L_2 \vee \dots \vee L_n$ wird $\{L_1, \dots, L_n\}$ verwendet):
 - **$(\text{Bat_OK} \wedge \text{Lifiable}) \Rightarrow \text{Moves}$**
 - **äquivalent zu: $\neg \text{Bat_OK} \vee \neg \text{Lifiable} \vee \text{Moves}$**
 - **äquivalent zu: $\{\neg \text{Bat_OK}, \neg \text{Lifiable}, \text{Moves}\}$**
- **NIL $\hat{=}$ $\{ \}$ $\hat{=}$ F (Wert immer falsch)**

10.2 Satz (Resolutionsregel)

- C_1 und C_2 seien Klauseln und λ ein Atom. Dann kann man aus $\{\lambda\} \cup C_1$ und $\{\neg \lambda\} \cup C_2$ schließen, dass $C_1 \cup C_2$ gilt. $C_1 \cup C_2$ heißt **Resolvente**.
- **Beweis:**

λ	C_1	C_2	$\{\lambda\} \cup C_1$	$\{\neg \lambda\} \cup C_2$	$C_1 \cup C_2$	$(\{\lambda\} \cup C_1) \wedge (\{\neg \lambda\} \cup C_2) \Rightarrow (C_1 \cup C_2)$
w	w	w	w	w	w	w
w	w	f	w	f	w	w
w	f	w	w	w	w	w
w	f	f	w	f	f	w
f	w	w	w	w	w	w
f	w	f	w	w	w	w
f	f	w	f	w	w	w
f	f	f	f	w	f	w

10.3 Beispiele

a) $\frac{R \vee P, \neg P \vee Q}{R \vee Q}$ steht für $(R \vee P, \neg P \vee Q) = R \vee Q$
 manchmal $(R \vee P, \neg P \vee Q) \text{ rel } R \vee Q$

Dieses Prinzip heißt wegen $(\neg R \Rightarrow P \wedge P \Rightarrow Q) \Rightarrow \neg R \Rightarrow Q$
 auch Verkettung.

b) $\frac{R, \neg R \vee P}{P}$, dieses Beweisschema ist der Modus
 Ponens $(R \wedge (R \Rightarrow P)) \Rightarrow P$

c) $\frac{P \vee Q \vee R \vee S, \neg P \vee Q \vee W}{Q \vee R \vee S \vee W}$, Q taucht nur einmal auf

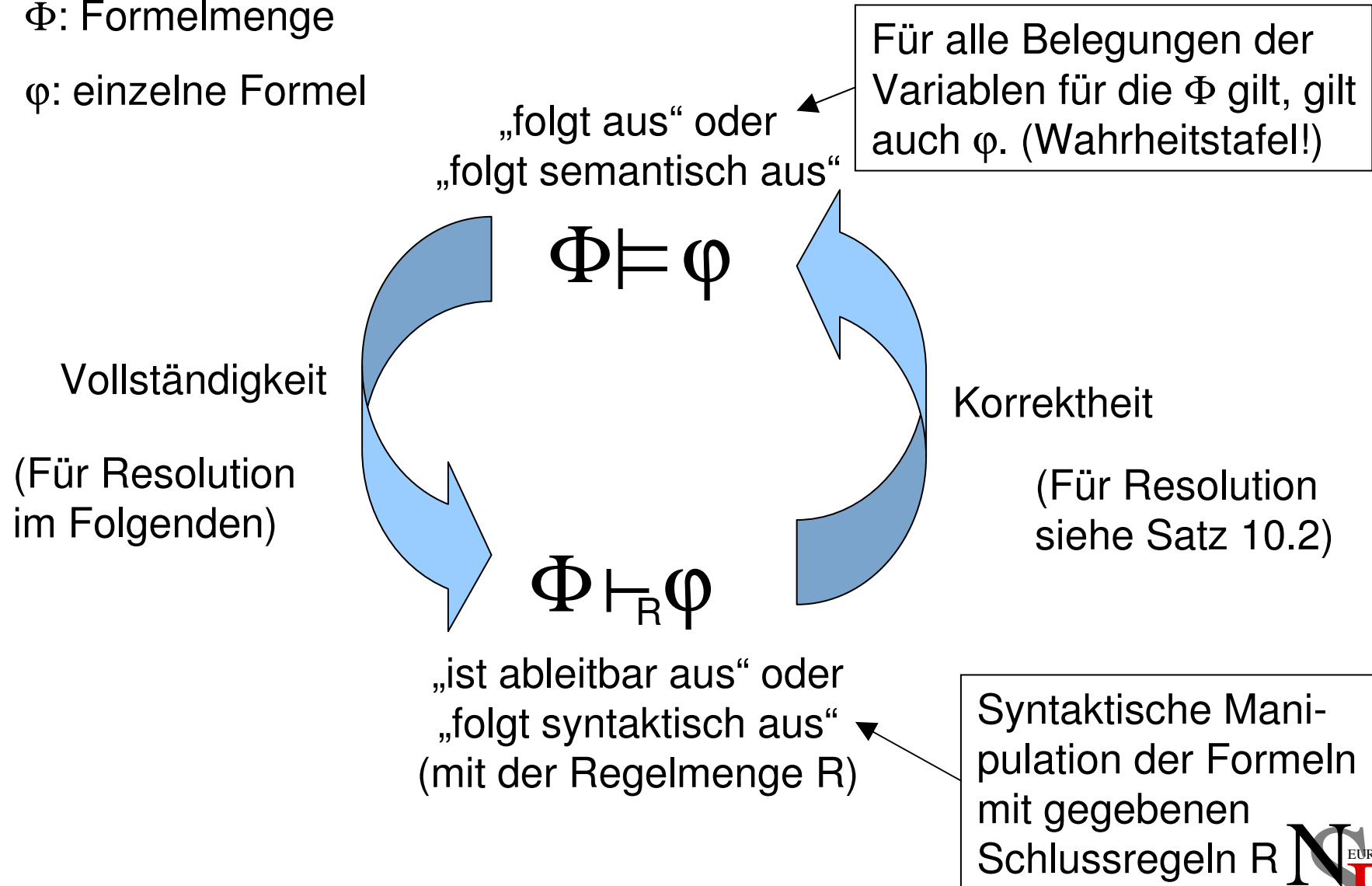
d) $\frac{P \vee \overset{\downarrow}{Q} \vee \neg R, P \vee W \vee \neg \overset{\downarrow}{Q} \vee R}{P \vee \neg R \vee R \vee W}, \frac{P \vee \overset{\downarrow}{Q} \vee \neg \overset{\downarrow}{R}, P \vee W \vee \neg \overset{\downarrow}{Q} \vee \overset{\downarrow}{R}}{P \vee Q \vee W \vee \neg Q},$

$P \vee W$ ist keine Resolvente

Korrektheit und Vollständigkeit

Φ : Formelmenge

φ : einzelne Formel



10.6 Bemerkung

Resolution ist korrekt, aber Resolution ist nicht vollständig:

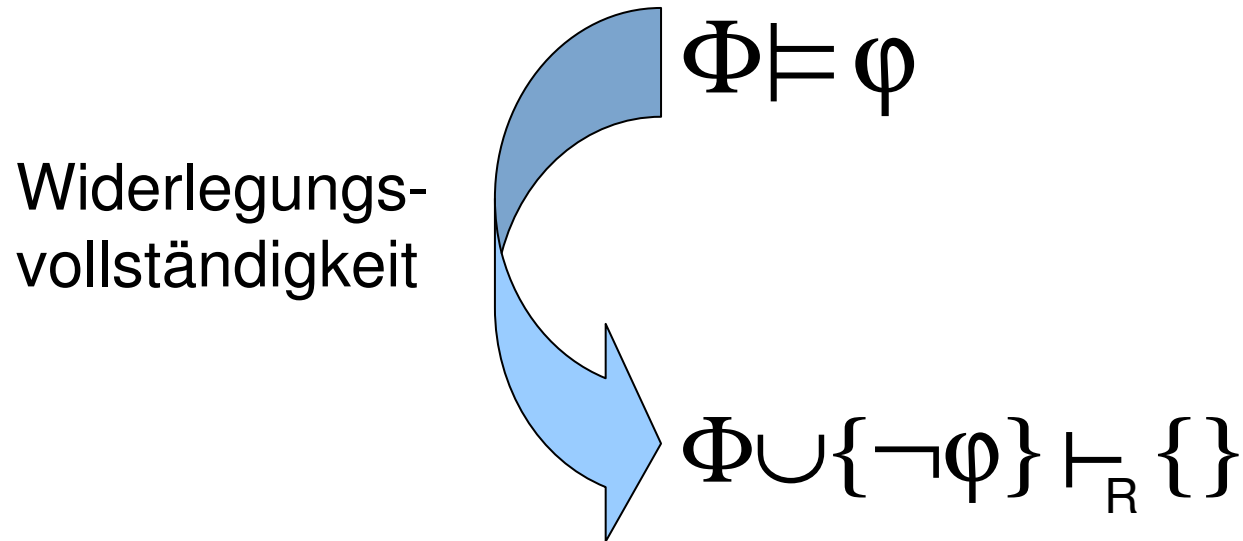
$P \wedge R \models P \vee R$ jedoch kann aus $\{P, R\}$ nichts mit dem Resolutionsverfahren geschlossen werden.

Ausweg: Wir zeigen, dass $\{P, R\}$ und $\neg\{P \vee R\}$ widersprüchlich sind (das kann man mit dem Resolutions-verfahren leicht zeigen):

$\neg(P \vee R) \text{ äq } \neg P \wedge \neg R$, also ist die Klauselmenge insgesamt $\{\neg P, \neg R, P, R\}$. Mit $res(\neg P, P) = \text{NIL} = F = \{ \}$ erhält man jedoch eine leere Klauselmenge und somit F.

Also muss das Gegenteil $P \wedge R \models P \vee R$ gelten.

Widerlegungsvollständigkeit



- Die Resolutionsregel ist nicht vollständig, aber widerlegungsvollständig (Beweis sehr komplex).
- Durch die Umformung vergrößert man die Menge der möglichen Schlüsse.
- Deshalb werden meist Widerspruchsbeweise geführt.

Umwandlung in Klauseln

10.4 Satz. Jede Aussage (well formed formula, wff) kann man in eine äquivalente Konjunktion von Klauseln umformen.

Anmerkung: Resolutionsregel führt nicht aus der Menge der Klauseln heraus!

10.5 Beispiel. (typische Vorgehensweise)

Gegeben: $\neg(P \rightarrow Q) \vee (R \rightarrow P)$

- a) **Implikation beseitigen:** $\neg(\neg P \vee Q) \vee (\neg R \vee P)$
- b) **Negation zu den Atomen bringen:** $(P \wedge \neg Q) \vee (\neg R \vee P)$
(*de Morgansche Regeln, $\neg\neg, \dots$*)
- c) **Konjunktive Normalform erzeugen:**
 $(P \vee \neg R) \wedge (\neg Q \vee \neg R \vee P)$ (*Distr., Assoz.,...*)
- d) **Klauselform erzeugen:**

$$\{(P \vee \neg R), (\neg Q \vee \neg R \vee P)\}$$

10.7 Beweis mit Resolutionsverfahren

Seien Δ eine Menge von Aussagen und ω eine Aussage.

Frage: Gilt $\Delta \models \omega$ (d.h. folgt ω aus Δ)?

Beweisprozedur:

Schritt 1: Konvertiere Δ in (konjunktive) Menge von Klauseln

Schritt 2: Konvertiere $\neg\omega$ in Klauselform

Schritt 3: Vereinige die Klauselmengen aus Schritt 1 und Schritt 2 in der Klauselmenge Γ

Schritt 4: Wende das Resolutionsprinzip auf die Klauseln in Γ an und füge die Ergebnisse zu Γ hinzu, bis die leere Klausel (der Widerspruch) erzeugt ist oder keine neuen Klauseln erzeugt werden können.

10.7 Beweis mit Resolventenverfahren

Anmerkungen:

Das Verfahren ist (widerlegungs-)vollständig:

Gilt $\Delta \models \omega$, dann wird die leere Klausel auch erzeugt.

Das Verfahren ist auch entscheidbar:

Ist $|\Delta| < \infty$ und gilt $\Delta \models \omega$ nicht, so terminiert das Verfahren.

10.8 Beispiel

$\Delta = \{\text{BAT_OK}, \neg\text{MOVES}, \text{BAT_OK} \wedge \text{LIFTABLE} \Rightarrow \text{MOVES}\}$

$\omega = \neg\text{LIFTABLE}$



Wissen: Falls die Batterie des Roboters OK ist und der Block nicht zu schwer ist, dann kann der Arm des Roboters den Block hochheben.

Frage: $\Delta \models \omega$?

- a) Semantische Lösung: **Mit Wahrheitstabellen nachweisen**
- b) Syntaktische Lösung: **Mit der obigen Beweisprozedur:**

Klauselmengen:

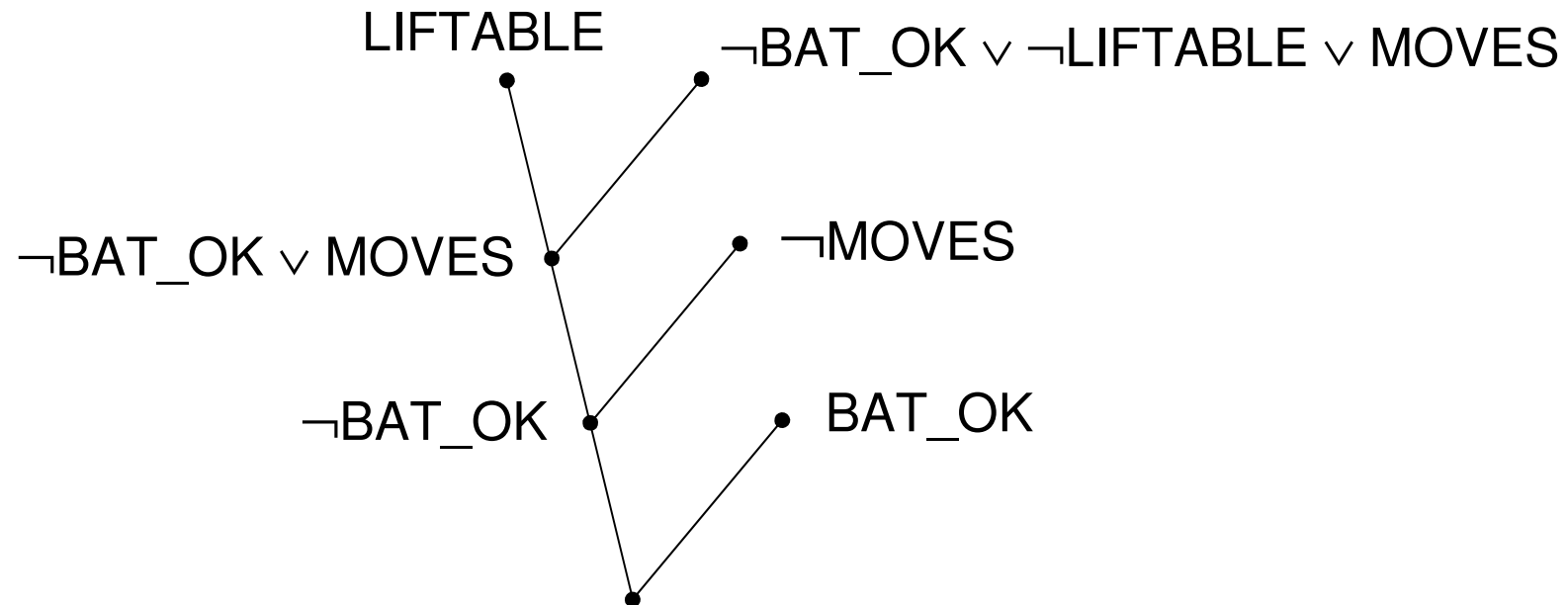
$\Delta = \{\text{BAT_OK}, \neg\text{MOVES},$
 $\neg\text{BAT_OK} \vee \neg\text{LIFTABLE} \vee \text{MOVES}\}$

$\neg\omega = \{\text{LIFTABLE}\}$

$\Gamma = \Delta \cup \neg\omega$

10.8 Beispiel

Widerspruchsbeweis durch Resolventenbildung:



Anmerkung: **Diese Beweise lassen sich gut automatisieren (Theorem-Beweiser, Maschinelles Beweisen) – siehe Internet für Programme.**

10.9 Suchstrategien in Widerlegungsbäumen

Problem:

Welche Resolvente wird zuerst gebildet um NIL schnell zu finden?

- a) Tiefensuche; Breitensuche; Klauselmengen der Klauseln mit nur einem Literal bevorzugen (unit clause strategy)**
- b) Verfeinerte Strategien wie set of support, linear input, ancestering filtering werden in Theorembeweisern für effiziente Implementierung verwendet**

Horn-Klauseln

10.10 Definition.

Eine Horn-Klausel ist eine Klausel, die höchstens ein positives Literal enthält.

10.11 Beispiel.

a) $P, \neg P \vee Q, \neg P \vee \neg Q \vee R, \neg P \vee \neg R$

b) Es gibt 3 Typen von Horn-Klauseln:

$$\rightarrow P \quad (\text{Fakt})$$

$$P_1 \wedge P_2 \wedge P_3 \rightarrow Q \quad (\text{Regel})$$

$$P_1 \wedge P_2 \rightarrow \quad (\text{Ziel, Anfrage})$$

10.12 Bemerkung.

Es gibt schnelle Deduktionsalgorithmen (in linearer Zeit) für Horn-Klauseln, deswegen werden (falls möglich) Horn Klauseln verwendet. (Horn-Klauseln haben keine Disjunktionen positiver Literale!)

Prädikatenkalkül (predicate calculus)

Anmerkung: Das Prädikatenkalkül ermöglicht es Individuen zu definieren (zu benennen) und ihnen Eigenschaften zuzuweisen. Dies ist im Aussagenkalkül nicht möglich.

10.13 Begriffsbildung

Terme: Vater_von(John); times(4, plus(3, 6)); Rolf; **x**; ...

Variable

Formeln:

Atomare Formel (Atom):

greater_than(7, 2); P(A,B,C,D); Q(**x**); ...

Zusammengesetzte Formel:

greater_than(7, 2) $\wedge$ less_than(14, 4) $\vee$
 $\neg$ brother(John, Sam) $\vee$ P

Formeln mit Quantoren:

$$\forall x (P(x) \rightarrow Q(x)), \exists x \forall y (R(x, y))$$

Prädikatenkalkül

Klauseln (Beispiel):

Im Folgenden ist die Funktion $.$ eine Abbildung $I \times I \rightarrow I$, wobei I Menge der Individuen ist (*append operator*).

$$(\text{liste}([]) \wedge \forall X \forall L : (\text{liste}(L) \Rightarrow \text{liste}(. (X, L)))) \Rightarrow \text{liste}(. (a, []))$$

Kann man auch als Programm schreiben:

$\Rightarrow \text{liste}([])$

(Fakten)

$\text{liste}(L) \Rightarrow \text{liste}(. (X, L))$

(Regeln)

$\text{liste}(. (a, [])) \Rightarrow$

(Anfragen)

(Die Formel ist allgemeingültig, das Programm beweist die Allgemeingültigkeit.)

Prädikatenkalkül

Resolventen:

Werden wie im Aussagenkalkül gebildet. Eine Besonderheit ist die geeignete Substitution von freien Variablen, damit die Resolventenbildung möglich ist:

$$P(f(y), A) \vee Q(B, C), \neg P(x, A) \vee R(x, C) \vee S(A, B)$$

Ersetze x durch $f(y)$, dann erhält man als Resolvente

$$Q(B, C) \vee R(f(y), C) \vee S(A, B)$$

Unifikation

10.14 Unifikation. Unifikation nennt man das Finden einer geeigneten Substitution um Atome „gleich“ zu machen und somit die Resolution anwenden zu können.

Beispiele von Substitutionen:

Klausel	Substitution	Ergebnis (Substitutionsinstanz)
$P[X, f(y), B]$	$s_1 = \{z \mid x, w \mid y\}$	$P[z, f(w), B]$ (alphabetische Variante)
$P[X, f(y), B]$	$s_2 = \{A \mid y\}$	$P[x, f(A), B]$
$P[X, f(y), B]$	$s_3 = \{g(z) \mid x, A \mid y\}$	$P[g(z), f(A), B]$
$P[X, f(y), B]$	$s_4 = \{C \mid x, A \mid y\}$	$P[C, f(A), B]$ (Grundinstanz)

Anmerkung: Sind mehrere Substitutionen in einer Menge zusammengefasst werden alle gleichzeitig ausgeführt.

Unifikation

- Ist ω ein Ausdruck und s eine Substitution, so ist ωs eine Substitutionsinstanz, z. B. $P(x, f(y), B)s_1 = P(Z, f(w), B)$

- Substitutionen können auch hintereinander geschaltet werden:

Sei $\omega = P(x, y)$, $s_1 = \{f(y)|x\}$, $s_2 = \{A|y\}$

dann erhält man:

$$\begin{aligned}(\omega s_1)s_2 &= [P(f(y), y)]s_2 = P(f(A), A) \\ &= [P(x, y)]\{f(A)|x, A|y\} = \omega(s_1 s_2)\end{aligned}$$

$$\omega(s_1 s_2) = P(f(A), A) \neq P(f(y), A) = \omega(s_2 s_1)$$

- Anmerkung: Bei der Hintereinanderschaltung von Substitutionen wird die rechts stehende Substitution zunächst auf die links stehende angewandt. Es gilt Assoziativität.

10.15 Unifizierbarkeit

Definition: Eine Menge $\{\omega_1, \dots, \omega_n\}$ von Ausdrücken heißt unifizierbar (verschmelzbar), wenn es eine Substitution s gibt, so dass $\omega_1 s = \omega_2 s = \dots = \omega_n s$ gilt. s heißt dann Unifizierer der Menge.

Die Menge $\{P(x, f(y), B), P(x, f(B), B)\}$ kann man z.B. mit der Substitution $s = \{A \mid x, B \mid y\}$ verschmelzen. Sie ist jedoch nicht die „einfachste“ Substitution, $s' = \{B \mid y\}$ ist einfacher.

Unifizierbarkeit

Definition: Es sei $A = \{w_1, \dots, w_n\}$ eine Menge von Ausdrücken. Dann gilt:

g ist der allgemeinste Unifizierer von $A \Leftrightarrow$

- (i) g ist Unifizierer von A
- (ii) Ist s Unifizierer von A , dann gibt es eine Substitution s' mit $As = Ags'$.

Die von dem allgemeinsten Verschmelzer produzierte Instanz ist bis auf alphabetische Varianten eindeutig.

Unifizierbarkeit

Anmerkung: Es gibt eine ganze Reihe von Algorithmen, um den allgemeinsten Verschmelzer zu finden. Der Algorithmus aus [Chang und Lee, 1973] nutzt Ausdrücke in Listenstruktur

$$\neg P(x, f(A, y)) \text{ entspricht } (\neg P\ x\ (f\ A\ y))$$

sowie die Idee der Abweichungsmenge (disagreement set).

Beispiel:

$\{\{A, z\}, \{y, B\}\}$ ist die Abweichungsmenge. Bei einer Implementierung bestimmt man in der Regel die Abweichungen nacheinander, d.h. die erste bestimmte Abweichung ist $\{A, z\}$ und die Substitution somit $\{A|z\}$.

$$\{(\neg P\ x\ (f\ A\ y)), (\neg P\ x\ (f\ z\ B))\}$$

10.16 Verschmelzungsalgorithmus

(Γ Menge von Ausdrücken in Listenstruktur)

(I) $k := 0, \Gamma_k := \Gamma, s_k := \varepsilon$ (ε leere Substitution)

(II) Enthält Γ_k nur identische Ausdrücke (d.h. als Menge nur einen Term), so ist s_k allgemeinsten Unifikator; Ende

(III) Man bestimme die Abweichungsmenge D_k von Γ_k

(IV) Falls es eine Variable v_k in D_k und einen Term t_k in D_k gibt, so dass v_k nicht in t_k vorkommt, so setze mit (V) fort. Andernfalls ist Γ nicht verschmelzbar.

(V) $s_{k+1} := s_k \{t_k \mid v_k\}, \Gamma_{k+1} := \Gamma_k \{t_k \mid v_k\}$ ($\Gamma_{k+1} = \Gamma s_{k+1}$)

(VI) $k := k+1$

(VII) Fahre mit (II) fort

Verschmelzungsalgorithmus

Beispiel:

- a) $\Gamma = \{p(a, x, f(g(y))), p(z, f(z), f(v))\}$
 $s_0 = \varepsilon, s_1 = \{a \mid z\}, \Gamma_1 = \{p(a, x, f(g(y))), p(a, f(a), f(v))\}$
 $s_2 = \{a \mid z, f(a) \mid x\}, \Gamma_2 = \{p(a, f(a), f(g(y))), p(a, f(a), f(v))\}$
 $s_3 = \{a \mid z, f(a) \mid x, g(y) \mid v\}, \Gamma_3 = \{p(a, f(a), f(g(y)))\}$
- b) **ist nicht unifizierbar!**

$$\Gamma = \{q(f(a), g(x)), q(y, y)\}$$

10.17 Resolution im Prädikatenkalkül

- γ_1, γ_2 seien zwei Klauseln (Mengen von Literalen). Gibt es ein Atom φ (Relationskonstante der Ordnung n , die von n Termen gefolgt wird und durch Kommata getrennt sind) in γ_1 und ein Literal $\neg\psi$ in γ_2 derart, dass φ und ψ den allgemeinsten Verschmelzer μ besitzen, so haben γ_1, γ_2 die Resolvente

$$\rho = [(\gamma_1 - \{\varphi\}) \cup (\gamma_2 - \{\neg\psi\})]\mu$$

- Dieses Verfahren ist korrekt und widerlegungsvollständig.
- Um Variablenkollisionen zu vermeiden, benennen wir die Variablen vorher so um, dass die Variablen in den Klauseln verschieden sind.

Resolution im Prädikatenkalkül

Beispiele:

a) $\{P(x), Q(x, y)\}$ und $\{\neg P(A), R(B, z)\}$

ergeben als Resolvente :

b) $\{Q(A, y), R(B, z)\}$

$\{P(x, x), Q(x), R(x)\}$ und $\{\neg P(A, z), \neg Q(B)\}$

ergeben als Resolventen

$\{Q(A), R(A), \neg Q(B)\}$ oder $\{P(B, B), R(B), \neg P(A, z)\}$

10.18 Überführung in Klauselform

Überführung von Formeln in Klauselform:

Schritt 1: $\Rightarrow, \Leftrightarrow$ eliminieren

$F \Rightarrow G$ entspricht $\neg F \vee G$

$F \Leftrightarrow G$ entspricht $F \Rightarrow G \wedge G \Rightarrow F$

Schritt 2: Gültigkeitsbereiche der Negationszeichen verkleinern.

Beispiel:

$\neg(F \vee G)$ entspricht $\neg F \wedge \neg G$

$\neg(\forall X : F)$ entspricht $\exists X : \neg F$

Überführung in Klauselform

Schritt 3: Quantoren nach vorne ziehen

Im Folgenden:

Z komme in $A(X)$ und $B(X)$ nicht vor

X komme in E nicht vor

$Q, Q_1, Q_2 \in \{\exists, \forall\}$

Mögliche Umformungen:

$(QX : A(X)) \wedge E$ äquivalent $QX : (A(X) \wedge E)$

$(QX : A(X)) \vee E$ äquivalent $QX : (A(X) \vee E)$

$\forall X : A(X) \wedge \forall X : B(X)$ äquivalent $\forall X : (A(X) \wedge B(X))$

$\exists X : A(X) \vee \exists X : B(X)$ äquivalent $\exists X : (A(X) \vee B(X))$

$Q_1(X) : A(X) \wedge Q_2(X) : B(X)$ äquivalent $Q_1 X : Q_2 Z : (A(X) \wedge B(Z))$

$Q_1(X) : A(X) \vee Q_2(X) : B(X)$ äquivalent $Q_1 X : Q_2 Z : (A(X) \vee B(Z))$

Überführung in Klauselform

Schritt 4: Den quantorenfreien Ausdruck als Konjunktion von Disjunktionen von Literalen darstellen:

$$(A \wedge B) \vee C \text{ äquivalent } (A \vee C) \wedge (B \vee C)$$

$$A \vee (B \wedge C) \text{ äquivalent } (A \vee B) \wedge (A \vee C)$$

Überführung in Klauselform

Beispiel zu Schritt 4:

$$\forall X : \forall Y : ((\exists Z : (p(X, Z) \vee p(Y, Z))) \Rightarrow (\exists Z : q(X, Y, Z)))$$

äquivalent

$$\forall X : \forall Y : (\neg(\exists Z : (p(X, Z) \vee p(Y, Z))) \vee \exists Z : q(X, Y, Z))$$

äquivalent

$$\forall X : \forall Y : ((\forall Z : (\neg p(X, Z) \wedge \neg p(Y, Z))) \vee \exists Z : q(X, Y, Z))$$

äquivalent

$$\forall X : \forall Y : \forall Z : \exists U : ((\neg p(X, Z) \wedge \neg p(Y, Z)) \vee q(X, Y, U))$$

äquivalent

$$\forall X : \forall Y : \forall Z : \exists U : ((\neg p(X, Z) \vee q(X, Y, U)) \wedge (\neg p(Y, Z) \vee q(X, Y, U)))$$

Überführung in Klauselform

Schritt 5: Elimination der Existenzquantoren:

Q sei der am weitesten links stehende Existenzquantor in **A**:

- Wenn vor **Q** kein Allquantor steht, streiche (QX) : aus **A** und ersetze **X** durch Konstante, die nicht in **A** vorkommt
- Sind $Q_1X_1, \dots, Q_mX_m$: alle Allquantoren die links von **Q** stehen, so wählt man ein Funktionssymbol f , das nicht in **A** vorkommt, definiert f als m -stellige Funktion, und ersetzt jedes Vorkommen von **X** durch $f(X_1, \dots, X_m)$ und streicht (QX) : im Ausdruck **A**.

Anmerkung: f wird auch Skolemfunktion genannt. Die erhaltene Form nennt sich auch Skolem-Normalform.

Überführung in Klauselform

Schritt 6: Darstellung als Klauselmenge

$$\forall X : (A(X) \wedge B(X)) \text{ äquivalent } (\forall X : A(X)) \wedge (\forall Y : B(Y))$$

Beispiel: (siehe nächste Folie)

Einschub: Es gilt der folgende Satz:

Satz. A Aussage, S Klauselform von A :

- a) A unerfüllbar $\Leftrightarrow S$ unerfüllbar
- b) A und S nicht notwendig äquivalent

Anmerkung: Fehlende Äquivalenz liegt an Skolem-Normalform:

Beispiel: $P(a) \vee P(b) \models \exists x P(x)$, aber i.A. gilt nicht $P(a) \vee P(b) \models P(s)$
(Skolem-Konstante s).

Überführung in Klauselform

Beispiel zu Schritt 6: (Fortsetzung von Beisp. zu Schritt 4)

Schritt 5 folgend wird U zunächst durch $f(X,Y,Z)$ ersetzt:

$$\forall X : \forall Y : \forall Z : (\neg p(X, Z) \vee q(X, Y, f(X, Y, Z))) \wedge \\ (\neg p(Y, Z) \vee q(X, Y, f(X, Y, Z)))$$

äquivalent

$$\forall X_1 : \forall Y_1 : \forall Z_1 : (\neg p(X_1, Z_1) \vee q(X_1, Y_1, f(X_1, Y_1, Z_1))) \wedge \\ \forall X_2 : \forall Y_2 : \forall Z_2 : (\neg p(Y_2, Z_2) \vee q(X_2, Y_2, f(X_2, Y_2, Z_2)))$$

äquivalent

$$\{\neg p(X_1, Z_1) \vee q(X_1, Y_1, f(X_1, Y_1, Z_1)), \neg p(Y_2, Z_2) \vee q(X_2, Y_2, f(X_2, Y_2, Z_2))\}$$

äquivalent

$$p(X_1, Z_1) \Rightarrow q(X_1, Y_1, f(X_1, Y_1, Z_1))$$

$$p(Y_2, Z_2) \Rightarrow q(X_2, Y_2, f(X_2, Y_2, Z_2))$$

10.19 Resolution in Theorem-Beweisern

Es werden Widerspruchsbeweise geführt.

Beispiel: $\text{package}(x) \hat{=} x \text{ ist Paket} \quad \hat{=} P(x)$

a) $\text{inroom}(x, y) \hat{=} x \text{ ist ein Raum } y \quad \hat{=} I(x, y)$
 $\text{smaller}(x, y) \hat{=} x \text{ ist kleiner als } y \quad \hat{=} S(x, y)$

b) **Alle Pakete in Raum 27 sind kleiner als alle Pakete in Raum 28**

$\hat{=} \forall x \forall y : ((\text{package}(x) \wedge (\text{package}(y) \wedge \text{inroom}(x, 27) \\ \wedge \text{inroom}(y, 28))) \Rightarrow \text{smaller}(x, y))$

$\hat{=} \neg P(x) \vee \neg P(y) \vee \neg I(x, 27) \vee \neg I(y, 28) \vee S(x, y)$

Resolution in Theorem-Beweisern

Beispiel (Fortsetzung):

c) Jedes Paket in Raum 27 ist kleiner als eines der Pakete in Raum 29:

$$\hat{=} \forall x : \exists y : (P(x) \wedge P(y) \wedge I(x,27) \wedge I(y,29) \Rightarrow S(x, y))$$

Anmerkung: Vorsicht bei linguistischen Mehrdeutigkeiten, es könnte auch gemeint gewesen sein:

$$\exists y : \forall x (P(x) \wedge P(y) \wedge I(x,27) \wedge I(y,29) \Rightarrow S(x, y))$$

Der Vorteil des PKI ist, dass es den Nutzer zwingt, solche Mehrdeutigkeiten aufzulösen.

Resolution in Theorem-Beweisern

Beispiel (Fortsetzung):

d) Paket A ist in Raum 27 oder in Raum 28, Paket B ist in Raum 27, Paket B ist nicht kleiner als Paket A:

$$P(A), P(B), I(A, 27) \vee I(A, 28), I(B, 27), \neg S(B, A)$$

e) Kann man aus den Eigenschaften b) und d) folgern, dass Paket A in Raum 27 ist?

Ja:

$$\Delta = \{ \neg P(x) \vee \neg P(y) \vee \neg I(x, 27) \vee \neg I(y, 28) \vee S(x, y),$$

$$P(A), P(B),$$

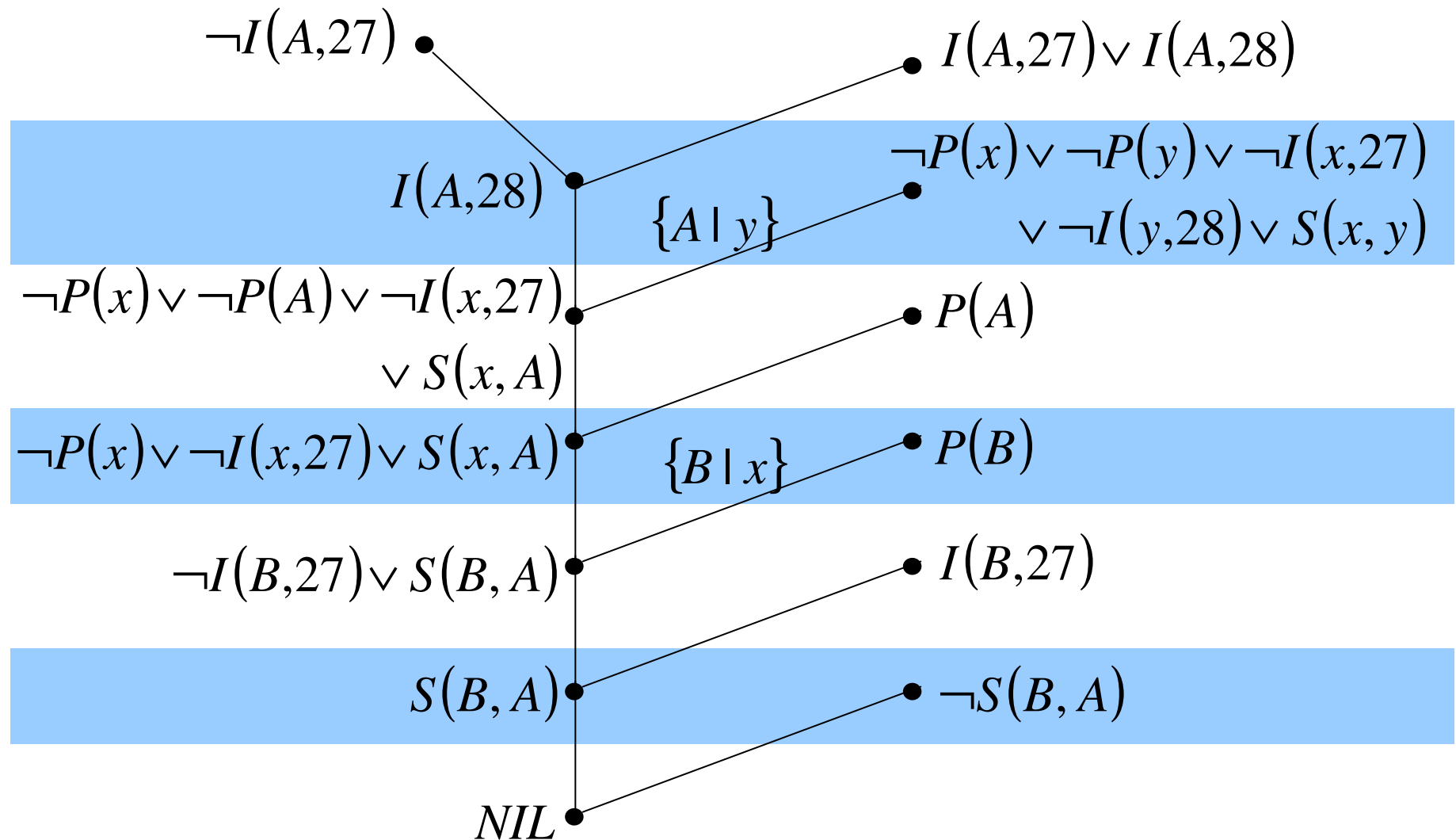
$$I(A, 27) \vee I(A, 28), I(B, 27),$$

$$\neg S(B, A) \}$$

$$w = I(A, 27)$$

$\Delta \cup \{ \neg w \}$ wird zum Widerspruch geführt :

Resolution in Theorem-Beweisern



Widerlegung durch Resolution

Resolution in Theorem-Beweisern

Beispiel (Fortsetzung):

f) Kann man aus c) und d) herausfinden, in welchem Raum A ist?

Anfrage: $\exists u : I(A, u)$

Lösungsidee:

Es wird ein Antwortliteral $Ans(u)$ eingeführt und dem Negat der Anfrage beigefügt:

$$\neg I(A, u) \vee Ans(u)$$

Dann wird die Resolution solange durchgeführt bis nur noch das Antwortliteral übrig bleibt.

Resolution in Theorem-Beweisern

