# Logische und funktionale Programmierung

## Vorlesung 2 und 3: Resolution

Babeş-Bolyai Universität, Department für Informatik, Cluj-Napoca
csacarea@cs.ubbcluj.ro

3. November 2017

# Herbrand-Struktur

Sei $(\Sigma, \Delta)$ eine Signatur. Eine Herbrand-Struktur (oder freie Struktur) hat dann die Gestalt $(T(\Sigma), \alpha)$, wobei für alle $f \in \Sigma_n$ mit $n \in \mathbb{N}$ gilt: $\alpha_f(t_1, \ldots, t_n) = f(t_1, \ldots, t_n)$. Falls eine Herbrand-Struktur Modell einer Formel $\phi$ ist, so bezeichnen wir sie auch als Herbrand-Modell von $\phi$.

# Beispiel

$$
\begin{aligned}
\alpha_n &= n \text{ für alle } n \in \mathbb{N} \\
\alpha_{\mathsf{monika}} &= \mathsf{monika} \\
\alpha_{\mathsf{karin}} &= \mathsf{karin} \\
&\vdots \\
\alpha_{\mathsf{datum}}(t_1, t_2, t_3) &= \mathsf{datum}(t_1, t_2, t_3) \text{ für alle } t_1, t_2, t_3 \in \mathcal{T}(\Sigma)
\end{aligned}
$$

# Beispiel

$$
\begin{aligned}
\alpha_{\mathsf{weiblich}} &= \{\mathsf{monika}, \mathsf{karin}, \ldots\} \\
\alpha_{\mathsf{maennlich}} &= \{\mathsf{werner}, \mathsf{klaus}, \ldots\} \\
\alpha_{\mathsf{mensch}} &= \mathcal{T}(\Sigma) \\
\alpha_{\mathsf{geboren}} &= \{(\mathsf{monika}, \mathsf{datum}(25, 7, 1972)), (\mathsf{werner}, \mathsf{datum}(12, 7, 1969)), \ldots\} \\
&\vdots
\end{aligned}
$$

*Diese Herbrand–Struktur kommt damit der intuitiv erwarteten Semantik des ursprünglichen Logikprogramms bereits wesentlich näher.*

# Erfüllbarkeitstest durch Herbrand-Strukturen

Der folgende Satz zeigt, dass man sich zur Untersuchung der Erfüllbarkeit in der Tat auf Herbrand-Strukturen beschränken kann.
Hierzu muss man die ursprüngliche Formel zunächst in eine erfüllbarkeitsäquivalente Formel in Skolem-Normalform überführen.

**Satz**

Sei $\Phi \subseteq F(\Sigma, \Delta, V)$ eine Menge von Formeln in Skolem-Normalform. Dann ist $\phi$ erfüllbar gdw. sie ein Herbrand-Modell besitzt.

Nachdem wir nun wissen, dass man zur Überprüfung der Unerfüllbarkeit einer Formel $\phi$ in Skolem-Normalform nur Herbrand-Strukturen betrachten muss, reduzieren wir dies nun auf die Überprüfung der Unerfüllbarkeit einer (unendlichen) Menge von Formeln ohne Variablen.
Dies entspricht der Überprüfung der Unerfüllbarkeit für eine (unendliche) Menge aussagenlogischer Formeln.
Die Idee hierzu besteht darin, alle in $\phi$ allquantifizierten Variablen mit allen möglichen Grundtermen zu instantiieren.
Auf diese Weise entsteht die Herbrand-Expansion von $\phi$.

# Herbrand Expansion

Sei $\phi \in F(\Sigma, \Delta, \mathcal{V})$ eine Formel in Skolem-Normalform, wobei $\phi = \forall X_1, \ldots, X_n \psi$ für eine quantorfreie Formel $\psi$. Die folgende Formelmenge $E(\phi)$ wird als Herbrand-Expansion von $\phi$ bezeichnet:

$$E(\phi) = \{\psi[X_1/t_1, \ldots, X_n/t_n] \mid t_1, \ldots, t_n \in T(\Sigma)\}.$$

Die Herbrand-Expansion einer Formel $\forall X_1, \ldots, X_n \psi$ ist also die Menge aller Grundinstanzen von $\psi$.

# Beispiel

Wir betrachten die Formel $\phi$:

$$\forall X \texttt{mutterVon}(\texttt{renate}, \texttt{susanne}) \wedge \neg \texttt{mutterVon}(\texttt{X}, \texttt{susanne})$$

deren Unerfüllbarkeit gezeigt werden muss, um nachzuweisen, dass die Anfrage

$$? - \texttt{mutterVon}(\texttt{X}, \texttt{susanne}).$$

aus dem Logikprogramm mit dem Faktum

$$\texttt{mutterVon}(\texttt{renate}, \texttt{susanne}).$$

folgt. Wir erhalten hier
$E(\phi) = \{mutterVon(renate, susanne) \wedge \neg mutterVon(karin, susanne),$
$mutterVon(renate, susanne) \wedge \neg mutterVon(renate, susanne),$
$mutterVon(renate, susanne) \wedge$
$\neg mutterVon(datum(karin, werner, 1982), susanne), \ldots\}$.

# Erfüllbarkeit der Herbrand-Expansion

Sei $\phi$ eine Formel in Skolem-Normalform. Dann ist $\phi$ erfüllbar gdw. $E(\phi)$ erfüllbar ist.

- Pradikatenlogische Formeln ohne Variablen entsprechen aussagenlogischen Formeln. Der Grund ist, dass man jede atomare Teilformel $p(t_1, \ldots, t_n)$ als aussagenlogische Variable ansehen kann, die entweder wahr oder falsch sein kann. Es genügt zu untersuchen, ob die Menge $E(\phi)$ ein Herbrand-Modell hat.
- Herbrand-Strukturen unterscheiden sich nur in der Deutung der Prädikatssymbole und bei variablenfreien Formeln bedeutet die Suche nach einem Herbrand-Modell nichts anderes als die Suche nach einer aussagenlogischen Belegung der aussagenlogischen Variablen $p(t_1, \ldots, t_n)$ mit wahr oder falsch.
- Die Frage nach der Unerfüllbarkeit einer Formel ist jetzt also reduziert worden auf die Suche nach einer erfüllenden Belegung für eine (im Allgemeinen unendliche) Menge aussagenlogischer Formeln.

# Beispiel

- Wir betrachten noch einmal die Herbrand-Expansion der Formel $\phi$ aus dem obigen Beispiel.
- Ersetzt man hier jede atomare Formel $p(t_1, \ldots, t_n)$ ohne Variablen durch eine aussagenlogische Variable $V_{p(t_1,\ldots,t_n)}$, so ergibt sich aus $E(\phi)$ die folgende unendliche Menge aussagenlogischer Formeln:
  $\{V_{mutterVon(renate,susanne)} \wedge \neg V_{mutterVon(karin,susanne)},$
  $V_{mutterVon(renate,susanne)} \wedge \neg V_{mutterVon(renate,susanne)},$
  $V_{mutterVon(renate,susanne)} \wedge \neg V_{mutterVon(datum(karin,werner,1982),susanne)},$
  $\ldots\}.$
- Die Formel $\phi$ und damit $E(\phi)$ ist nun genau dann erfüllbar, wenn es eine Belegung der aussagenlogischen Variablen $V_{p(t_1,\ldots,t_n)}$ mit wahr oder falsch gibt, die die obige aussagenlogische Formelmenge erfüllt.

# BEISPIEL

- Natürlich gibt es solch eine Belegung nicht, denn die Formel
  $V_{mutterVon(renate,susanne)} \wedge \neg V_{mutterVon(renate,susanne)}$
  ist weder für die Belegung von $V_{mutterVon(renate,susanne)}$ mit wahr noch mit falsch erfüllt.

# Algorithmus von Gilmore

- Eingabe: Formeln $\phi_1, \ldots, \phi_k, \phi \in F(\Sigma, \Delta, \mathcal{V})$
- Ziel: herauszufinden, ob $\{\phi_1, \ldots, \phi_k\} \models \phi$ gilt.

# Algorithmus von Gilmore

- Sei $\psi$ die Formel $\phi_1 \wedge \cdots \wedge \phi_k \wedge \phi$. Es gilt $\{\phi_1, \ldots, \phi_k\} \models \phi$ gdw. $\psi$ unerfüllbar ist.
- Überführe $\psi$ in Skolem-Normalform. Die entstehende Formel ist erfüllbarkeitsäquivalent zur ursprünglichen Formel.
- Wähle eine Aufzählung $\{\psi_1, \psi_2, \ldots\} = E(\forall\psi)$ der Herbrand-Expansion von $\forall\psi$ und ersetze dabei alle atomaren Formeln durch aussagenlogische Variablen.
- $E(\forall\psi)$ ist genau dann (aussagenlogisch) erfüllbar, wenn $\forall\psi$ erfüllbar ist. Nach dem Kompaktheitssatz der Aussagenlogik ist dies genau dann der Fall, wenn alle endlichen Teilmengen von $E(\forall\psi)$ aussagenlogisch erfüllbar sind.

# Algorithmus von Gilmore

- Prüfe, ob $\psi_1, \psi_1 \wedge \psi_2, \psi_1 \wedge \psi_2 \wedge \psi_3, \ldots$ aussagenlogisch erfüllbar sind (indem man alle möglichen Wahrheitsbelegungen der vorkommenden aussagenlogischen Variablen durchtestet). Sofern eine dieser Formeln nicht erfüllbar ist, brich ab und gib true zurück.

# BEMERKUNG

- Dieser Algorithmus terminiert und liefert genau dann das Ergebnis true zurück, wenn $\{\phi_1, \ldots, \phi_k\} \models \phi$ gilt.
- Sofern dies nicht gilt, terminiert der Algorithmus nicht.
- Der Nachteil des Algorithmus ist allerdings, dass er noch nicht sehr zielgerichtet vorgeht und daher noch sehr ineffizient ist.

# Grundresolution

## Konjunktive Normalform, Literal, Klausel

### Definition

*Eine Formel $\psi$ ist in konjunktiver Normalform (KNF) gdw. sie quantorfrei ist und die folgende Gestalt hat.*

$$(L_{1,1} \vee \cdots \vee L_{1,n_1}) \ldots (L_{m,1} \vee \cdots \vee L_{m,n_m}).$$

*Hierbei sind die $L_{i,j}$ Literale, d.h., es sind atomare oder negierte atomare Formeln der Gestalt $p(t_1, \ldots, t_k)$ oder $\neg p(t_1, \ldots, t_k)$. Zu einem Literal L definieren wir sein Negat $\overline{L}$ wie folgt:*

$$\overline{L} = \begin{cases} A & fallsL = A \in At(\Sigma, \Delta, \mathcal{V}) \\ A & fallsL = \neg AmitA \in At(\Sigma, \Delta, \mathcal{V}) \end{cases}$$

Eine Menge von Literalen wird als Klausel bezeichnet. Jede Formel $\psi$ in KNF wie oben entspricht der zugehörigen Klauselmenge

$$K(\psi) = \{\{L_{1,1}, \ldots, L_{1,n_1}\}, \ldots, \{L_{m,1}, \ldots, L_{m,n_m}\}\}.$$

Eine Klausel steht für die allquantifizierte Disjunktion ihrer Literale und eine Klauselmenge repräsentiert die Konjunktion ihrer Klauseln. Wir sprechen daher im Folgenden auch von Erfüllbarkeit und Folgerbarkeit von Klauselmengen. Falls nichts explizit anderes gesagt wird, betrachten wir im Folgenden stets nur endliche Klauselmengen. Die leere Klausel wird oft als $\square$ geschrieben und sie ist nach Definition unerfüllbar.

# Überführung in KNF

**Satz**

Zu jeder quantorfreien Formel $\psi$ lässt sich automatisch eine Formel $\psi'$ in konjunktiver Normalform konstruieren, so dass $\psi$ und $\psi'$ äquivalent sind.

**Beweis.** Zunächst werden alle Teilformeln $\psi_1 \leftrightarrow \psi_2$ in $\psi$ durch $\psi_1 \rightarrow \psi_2 \wedge \psi_2 \rightarrow \psi_1$ ersetzt. Anschließend werden alle Teilformeln $\psi_1 \rightarrow \psi_2$ in $\psi$ durch $\neg\psi_1 \vee \psi_2$ ersetzt. Die Überführung der verbleibenden Formel geschieht mit folgendem Algorithmus $KNF$, dessen Terminierung und Korrektheit offensichtlich ist. Der Algorithmus bekommt als Eingabe eine beliebige quantorfreie Formel $\psi$ ohne die Junktoren "$\leftrightarrow$" und "$\rightarrow$" und liefert als Ausgabe eine dazu äquivalente Formel in KNF.

- Falls $\psi$ atomar ist, so gib $\psi$ zurück.
- Falls $\psi = \psi_1 \wedge \psi_2$ ist, so gib $KNF(\psi_1) \wedge KNF(\psi_2)$ zurück.
- Falls $\psi = \psi_1 \vee \psi_2$ ist, so berechne $KNF(\psi_1) = \bigwedge_{i\in\{1,\dots,m_1\}} \psi_i'$ und $KNF(\psi_2) = \bigwedge_{j\in\{1,\dots,m_2\}} \psi_j''$. Hierbei sind $\psi_i'$ und $\psi_i''$ Disjunktionen von Literalen. Durch Anwendung des Distributivgesetzes erhält man die Formel $\bigwedge_{i\in\{1,\dots,m_1\}, j\in\{1,\dots,m_2\}} \psi_i' \vee \psi_j''$ in KNF, die zurückgegeben wird.
- Falls $\psi = \neg\psi_1$ ist, so berechne $KNF(\psi_1) = \bigwedge_{i\in\{1,\dots,m\}}(\bigvee_{j\in\{1,\dots,n_i\}} L_{i,j})$. Mehrfache Anwendung der de Morganschen Regel auf $\neg \bigwedge_{i\in\{1,\dots,m\}}(\bigvee_{j\in\{1,\dots,n_i\}} L_{i,j})$ liefert $\bigvee_{i\in\{1,\dots,m\}}(\bigwedge_{j\in\{1,\dots,n_i\}} \overline{L_{i,j}})$. Anschließend geht man wie im vorigen Fall vor und wendet das Distributivgesetz an. Dies ergibt die Formel $\bigwedge_{j_1\in\{1,\dots,n_1\},\dots,j_m\in\{1,\dots,n_m\}} \overline{L_{1,j_1}} \vee \ldots \vee \overline{L_{m,j_m}}$ in KNF, die zurückgegeben wird.

$\square$

# Aussagenlogische Resolution

## Definition

*Seien $K_1$ und $K_2$ variablenfreie Klauseln. Dann ist die Klausel $R$ Resolvent von $K_1$ und $K_2$ gdw. es ein Literal $L \in K_1$ gibt mit $\neg L \in K_2$ und $R = (K1 \setminus \{L\}) \cup (K_2 \setminus \{\neg L\})$. Für eine Klauselmenge $K$ definieren wir*
$Res(K) = K \cup \{R \mid R \text{ ist Resolvent zweier Klauseln aus } K\}$.
*Weiterhin sei $Res_0(K) = K$ und $Res_{n+1}(K) = Res(Resn(K))$ für alle $n \geq 0$. Schließlich definieren wir die Menge der aus $K$ durch Resolution herleitbaren Klauseln als*

$$Res^*(K) = \bigcup_{n \geq 0} Res_n(K).$$

Offensichtlich gilt $\Box \in Res^*(\mathcal{K})$ gdw. es eine Folge von Klauseln $K_1, \ldots, K_m$ gibt, so dass $K_m = \Box$ ist und so dass für alle $1 \leq i \leq m$ gilt:

- $K_i \in \mathcal{K}$ oder
- $K_i$ ist ein Resolvent von $K_j$ und $K_k$ für $j, k < i$

Zur Darstellung von Resolutionsbeweisen schreiben wir oftmals das folgende Diagramm, um deutlich zu machen, dass $R$ durch Resolution aus $K_1$ und $K_2$ entsteht.

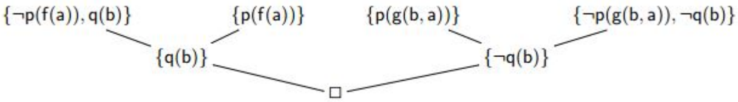
$$\begin{tikzcd}
K_1 \arrow[dr, no head] & & K_2 \arrow[dl, no head] \\
& R &
\end{tikzcd}$$

# Beispiel

$\Delta_1 = \{p, q\}, \Sigma_0 = \{a, b\}, \Sigma_1 = \{f\}$ UND $\Sigma_2 = \{g\}$

$$
\begin{aligned}
K_1 &= \{\neg p(f(a)), q(b)\} \\
K_2 &= \{p(f(a))\} \\
K_3 &= \{p(g(b,a))\} \\
K_4 &= \{\neg p(g(b,a)), \neg q(b)\}
\end{aligned}
$$

*Es gilt:*



# Aussagenlogisches Resolutionslemma

Sei $\mathcal{K}$ eine Menge von variablenfreien Klauseln. Falls $K_1, K_2 \in \mathcal{K}$ und $R$ Resolvent von $K_1$ und $K_2$ ist, dann sind $\mathcal{K}$ und $\mathcal{K} \cup \{R\}$ äquivalent.

**Beweis.** Jede Struktur $S$, die $\mathcal{K} \cup \{R\}$ erfüllt, erfüllt auch $\mathcal{K}$ (denn eine Klauselmenge repräsentiert die *Konjunktion* ihrer Klauseln). Somit gilt $\mathcal{K} \cup \{R\} \models \mathcal{K}$.

Umgekehrt sei nun $S$ eine Struktur, die $\mathcal{K}$ erfüllt. Es existiert ein Literal $L \in K_1$ mit $\overline{L} \in K_2$ und $R = (K_1 \setminus \{L\}) \cup (K_2 \setminus \{\overline{L}\})$. Wir nehmen an, dass $S \not\models \mathcal{K} \cup \{R\}$ gilt. Aus $S \models \mathcal{K}$ folgt dann $S \not\models R$. Falls $S \models L$, so folgt aus $S \models K_2$ auch $S \models K_2 \setminus \{\overline{L}\}$ und damit $S \models R$. Falls $S \models \overline{L}$, so folgt aus $S \models K_1$ auch $S \models K_1 \setminus \{L\}$ und damit $S \models R$. Somit gilt auch $\mathcal{K} \models \mathcal{K} \cup \{R\}$. □

Nun können wir die Korrektheit und Vollständigkeit des aussagenlogischen Resolutionskalküls beweisen. Hierbei bedeutet Korrektheit, dass man mit der Resolution keine falschen Aussagen beweisen kann (d.h., dass die Klauselmenge tatsächlich unerfüllbar ist, wenn man die leere Klausel durch Resolution herleiten kann).
Vollständigkeit bedeutet, dass alle wahren Aussagen durch Resolution beweisbar sind (d.h., dass man die leere Klausel aus jeder unerfüllbaren Klauselmenge durch Resolution herleiten kann).

# Korrektheit und Vollständigkeit der aussagenlogischen Resolution

**Satz**

Sei $\mathcal{K}$ eine möglicherweise unendliche Menge von variablenfreien Klauseln. Dann ist $\mathcal{K}$ unerfüllbar gdw. $\square \in Res^*(\mathcal{K})$.

- Nun können wir den Algorithmus von Gilmore zum Grundresolutionsverfahren verbessern.
- Wie in Schritt 1 und 2 des Algorithmus von Gilmore überführen wir das Folgerbarkeitsproblem zunächst in das Problem, die Unerfüllbarkeit einer Formel $\forall X_1, \ldots, X_n \psi$ in SkolemNormalform zu zeigen.
- Anschließend gehen wir wie folgt vor:

# Grundresolutionsalgorithmus

- Überführe $\psi$ in KNF bzw. in die entsprechende Klauselmenge $\mathcal{K}(\psi)$.
- Wähle eine Aufzählung $\{K_1, K_2, \ldots\}$ aller Grundinstanzen der Klauseln aus $\mathcal{K}(\psi)$. Dies entspricht der HerbrandExpansion von $\mathcal{K}(\psi)$, d.h., der Klauselmenge $\{\sigma(K) \mid K \in \mathcal{K}(\psi), \sigma$ ist Grundsubstitution, d.h., $\mathcal{V}(\sigma(K)) = \emptyset\}$. Hierbei bedeutet die Anwendung einer Substitution auf eine Klausel, dass die Substitution auf alle Literale der Klausel angewendet wird.
- Berechne $Res^*(\{K_1\}), Res^*(\{K_1, K_2\}), Res^*(\{K_1, K_2, K_3\}), \ldots$ Sofern eine dieser Mengen die leere Klausel $\Box$ enthält, brich ab und gib true zurück.

# Beispiel

- Wir wollen die Unerfullbarkeit der folgenden Formel in SkolemNormalform zeigen, deren quantorfreier Teil $\psi$ bereits in KNF vorliegt:

$$\forall X, Y (\neg p(X) \vee \neg p(f(a)) \vee q(Y)) \wedge p(Y) \wedge (\neg p(g(b, X)) \vee \neg q(b))$$

- Damit erhält man die folgende Klauselmenge $\mathcal{K}(\psi)$:

$$\{\{\neg p(X), \neg p(f(a)), q(Y)\}, \{p(Y)\}, \{\neg p(g(b, X)), \neg q(b)\}\}$$

- Wir wählen nun eine Aufzählung $\{K_1, K_2, K_3, K_4, \ldots\}$ der Grundinstanzen der drei Klauseln, wobei $K_1, \ldots, K_4$ die variablenfreien Klauseln sind. Wie im obigen Beispiel, kann man dann die leere Klausel herleiten und somit auch die Unerfüllbarkeit der ursprunglichen Formel zeigen. Man erkennt, dass hierbei sowohl $K_2$ als auch $K_3$ unterschiedliche Instanzen derselben Klausel $\{p(Y)\}$ sind.

# Beispiel Fortsetzung

- Dies kann nötig sein, um die Unerfüllbarkeit beweisen zu können.
- Ebenso erkennt man auch, dass durch die Instanzenbildung mehrere Literale in einer Klausel gleich gemacht werden können. So ist $K_1$ eine Instanz der Klausel $\{\neg p(X), \neg p(f(a)), q(Y)\}$, die aber nur noch zwei Literale enthält, da durch die Substitution von $X$ durch $f(a)$ die ersten beiden Literale der Klausel gleich gemacht werden.

# Korrektheit und Vollst andigkeit des Grundresolutionsalgorithmus

**Satz**

1. Falls eine Klauselmenge $\mathcal{K}$ unerfüllbar ist, dann existiert eine endliche Menge von Grundinstanzen von Klauseln aus $\mathcal{K}$ (d.h. eine endliche Teilmenge von $\{\sigma(K) \mid K \in \mathcal{K}(\psi), \sigma \text{ ist Grundsubstitution}\}$), die ebenfalls unerfüllbar ist.
2. Sei $\forall X_1, \ldots, X_n \psi$ eine Formel in SkolemNormalform, wobei $\psi$ in KNF ist. Dann ist $\forall X_1, \ldots, X_n \psi$ unerfüllbar gdw. es eine Folge von Klauseln $K_1, \ldots, K_m$ gibt, so dass $K_m = \square$ ist und so dass für alle $1 \leq i \leq m$ gilt:
   - $K_i$ ist Grundinstanz einer Klausel aus $\mathcal{K}(\psi)$ oder
   - $K_i$ ist ein Resolvent von $K_j$ und $K_k$ für $j, k < i$

- Der Nachteil des Algorithmus ist allerdings, dass die Suche nach geeigneten Grundinstanzen noch nicht zielgerichtet verläuft und man daher auf sehr ineffiziente Weise alle Grundinstanzen durchprobieren muss.
- Es müssen z.B. Entscheidungen für einige der Grundsubstitutionen in *vorausschauender Weise* getroffen werden, um spätere Resolutionen zu ermöglichen. Dies legt eine Modifikation nahe, bei der man die Substitutionen *zurückhaltend* wählt und immer nur insoweit Substitutionen durchführt, wie dies für den direkt nachfolgenden Resolutionsschritt benötigt wird.
- Dazu erlaubt man jetzt statt Grundsubstitutionen auch Substitutionen mit beliebigen Termen. Dies führt zur prädikatenlogischen Resolution.

# Beispiel

$p, q \in \Delta_1, f \in \Sigma_1$ UND $a \in \Sigma_0$

$$\{\ \{\mathsf{p}(X), \neg\mathsf{q}(X)\},\ \{\neg\mathsf{p}(\mathsf{f}(Y))\},\ \{\mathsf{q}(\mathsf{f}(\mathsf{a}))\}\ \}$$

*Um die ersten beiden Klauseln zu resolvieren, benutzen wir nun die Substitution* $\{X/\mathsf{f}(Y)\}$. *Nach Anwendung dieser Substitution ergeben sich die Literale* $\mathsf{p}(X)[X/\mathsf{f}(Y)] = \mathsf{p}(\mathsf{f}(Y))$ *und* $\neg\mathsf{p}(\mathsf{f}(Y))[X/\mathsf{f}(Y)] = \neg\mathsf{p}(\mathsf{f}(Y))$, *die zueinander komplementär sind. Die Substitution* $\{X/\mathsf{f}(Y)\}$ *ist daher ein Unifikator von* $\{\mathsf{p}(X), \mathsf{p}(\mathsf{f}(Y))\}$. *Der Resolvent ist dann die Klausel* $\{\neg\mathsf{q}(X)[X/\mathsf{f}(Y)]\} = \{\mathsf{q}(\mathsf{f}(Y))\}$ *mit dem verbleibenden zweiten Literal der ersten Klausel.*

*Der Vorteil ist, dass durch die Substitution* $\{X/\mathsf{f}(Y)\}$ *noch nicht festgelegt wurde, wie man* $Y$ *anschließend instantiiert. Es zeigt sich, dass man anschließend die Substitution* $\{Y/\mathsf{a}\}$ *verwenden muss, um schließlich die leere Klausel herzuleiten. Dies muss man aber*

# Beispiel

$p, q \in \Delta_1, f \in \Sigma_1$ UND $a \in \Sigma_0$

*nicht von vornherein erkennen* (d.h., *man muss nicht gleich die Substitution* $\{X/\mathsf{f}(\mathsf{a})\}$ *verwenden*), *sondern man benutzt jeweils nur Substitutionen wie* $\{X/\mathsf{f}(Y)\}$, *die so allgemein wie möglich sind. Mit anderen Worten, wir wählen im ersten Resolutionsschritt den allgemeinsten Unifikator von* $\{\mathsf{p}(X), \mathsf{p}(\mathsf{f}(Y))\}$ *und nicht den weniger allgemeinen Unifikator* $\{X/\mathsf{f}(\mathsf{a})\}$.

# Unifikation

## Definition

*Eine Klausel $K = \{L_1, \ldots, L_n\}$ ist unifizierbar gdw. es eine Substitution $\sigma$ mit $\sigma(L_1) = \cdots = \sigma(L_n)$ gibt (d.h., $\mid \sigma(\mathcal{K}) \mid = 1$). Solch eine Substitution heißt ein Unifikator von $\mathcal{K}$. Ein Unifikator $\sigma$ heißt allgemeinster Unifikator (most general unifier, mgu), falls es für jeden Unifikator $\Sigma_0$ eine Substitution $\delta$ gibt mit $\Sigma_0(X) = \delta(\sigma(X))$ für alle $X \in \mathcal{V}$.*

- Falls eine Klausel unifizierbar ist, so existiert auch ein allgemeinster Unifikator, der bis auf Variablenumbenennungen eindeutig ist.

# UNIFIKATIONSALGORITHMUS

1. Sei $\sigma = \varnothing$ die leere (oder "identische") Substitution.

2. Falls $|\sigma(K)| = 1$ ist, dann brich ab und gib $\sigma$ als mgu von $K$ aus.

3. Sonst durchsuche alle $\sigma(L_i)$ parallel von links nach rechts, bis in zwei Literalen die gelesenen Zeichen verschieden sind.

4. Falls keines der beiden Zeichen eine Variable ist, dann brich mit Clash Failure ab.

5. Sonst sei $X$ die Variable und $t$ der Teilterm im anderen Literal (hierbei kann $t$ auch eine Variable sein). Falls $X$ in $t$ vorkommt, dann brich mit Occur Failure ab. (Diese Überprüfung bezeichnet man als Occur Check.)

6. Sonst setze $\sigma = \{X/t\} \circ \sigma$ und gehe zurück zu Schritt 2.

# Beispiel

## $q \in \Delta_1, p \in \Delta_2, f, g \in \Sigma_2, h \in \Sigma_1$ und $a \in \Sigma_0$

$$\{ \mathsf{q}(\mathsf{f}(X, Y)), \ \mathsf{q}(\mathsf{g}(X, Y)) \}.$$

An der ersten Stelle, an der sich die beiden Literale unterscheiden, steht einmal das Funktionssymbol $\mathsf{f}$ und einmal das Funktionssymbol $\mathsf{g}$. Für keine Instantiierung der Variablen $X$ und $Y$ werden diese beiden Literale gleich, denn die Instantiierungen können diese verschiedenen Funktionssymbole nicht ändern. Diese Klausel ist daher nicht unifizierbar.

Als Beispiel für einen Occur Failure betrachten wir die folgende Klausel:

$$\{ \mathsf{q}(X), \ \mathsf{q}(\mathsf{h}(X)) \}.$$

An der ersten unterschiedlichen Stelle steht einmal die Variable $X$ und einmal der Term $\mathsf{h}(X)$, der die Variable $X$ enthält. Für keine Instantiierung von $X$ können diese Terme bzw. die ursprünglichen Literale gleich gemacht werden. Die Klausel ist daher ebenfalls nicht unifizierbar.

# Beispiel

$q \in \Delta_1, p \in \Delta_2, f, g \in \Sigma_2, h \in \Sigma_1$ UND $a \in \Sigma_0$

*Schließlich wenden wir den Unifikationsalgorithmus auf die Klausel aus den folgenden zwei Literalen an:*

$$\begin{array}{l} \neg \mathsf{p}(\mathsf{f}(Z, \mathsf{g}(\mathsf{a}, Y)), \mathsf{h}(Z)) \\ \neg \mathsf{p}(\mathsf{f}(\mathsf{f}(U, V), W), \mathsf{h}(\mathsf{f}(\mathsf{a}, Y))) \\ \quad\;\;\uparrow \end{array}$$

*Der Pfeil deutet die erste Stelle an, an der sich die beiden Literale unterscheiden. Somit ergibt sich* $\sigma = \{Z/\mathsf{f}(U, V)\}$. *Nun wendet man den bereits gefundenen Teilunifikator* $\sigma$ *auf die Literale an:*

$$\begin{array}{l} \neg \mathsf{p}(\mathsf{f}(\mathsf{f}(U, V), \mathsf{g}(\mathsf{a}, Y)), \mathsf{h}(\mathsf{f}(U, V))) \\ \neg \mathsf{p}(\mathsf{f}(\mathsf{f}(U, V), W), \mathsf{h}(\mathsf{f}(\mathsf{a}, Y))) \\ \qquad\qquad\;\;\uparrow \end{array}$$

# BEISPIEL

$q \in \Delta_1, p \in \Delta_2, f, g \in \Sigma_2, h \in \Sigma_1$ UND $a \in \Sigma_0$

*Es ergibt sich* $\sigma = \{W/\mathsf{g}(\mathsf{a}, Y)\} \circ \sigma = \{Z/\mathsf{f}(U, V), W/\mathsf{g}(\mathsf{a}, Y)\}$. *Durch Anwendung von* $\sigma$ *erhält man:*

$$\begin{array}{l} \neg\mathsf{p}(\mathsf{f}(\mathsf{f}(U, V), \mathsf{g}(\mathsf{a}, Y)), \mathsf{h}(\mathsf{f}(U, V))) \\ \neg\mathsf{p}(\mathsf{f}(\mathsf{f}(U, V), \mathsf{g}(\mathsf{a}, Y)), \mathsf{h}(\mathsf{f}(\mathsf{a}, Y))) \\ \qquad\qquad\qquad\qquad\qquad\quad\ \uparrow \end{array}$$

*Nun ergibt sich* $\sigma = \{U/\mathsf{a}\} \circ \sigma = \{Z/\mathsf{f}(\mathsf{a}, V), W/\mathsf{g}(\mathsf{a}, Y), U/\mathsf{a}\}$. *Die Anwendung von* $\sigma$ *führt zu:*

$$\begin{array}{l} \neg\mathsf{p}(\mathsf{f}(\mathsf{f}(\mathsf{a}, V), \mathsf{g}(\mathsf{a}, Y)), \mathsf{h}(\mathsf{f}(\mathsf{a}, V))) \\ \neg\mathsf{p}(\mathsf{f}(\mathsf{f}(\mathsf{a}, V), \mathsf{g}(\mathsf{a}, Y)), \mathsf{h}(\mathsf{f}(\mathsf{a}, Y))) \\ \qquad\qquad\qquad\qquad\qquad\qquad\ \uparrow \end{array}$$

# Beispiel

$q \in \Delta_1, p \in \Delta_2, f, g \in \Sigma_2, h \in \Sigma_1$ und $a \in \Sigma_0$

Schließlich erhält man $\sigma = \{Y/V\} \circ \sigma = \{Z/\mathsf{f}(\mathsf{a}, V), W/\mathsf{g}(\mathsf{a}, V), U/\mathsf{a}, Y/V\}$. Danach sind die beiden instantiierten Literale identisch, so dass dies der gesuchte mgu ist.