Ergänzung: Wozu `not` in PROLOG?

Beispiel: Prozedur zur Vereinigung (ohne Doubletten!) zweier Listen (hier: nur erstes Argument auf Doubletten prüfen)

```
uni([],L) :- write(L).
uni([H|T],L) :- member(H,L), uni(T,L).
uni([H|T],L) :- not(member(H,L)), uni(T,[H|L]).
```

Beispiel:

```
?- uni([s,d,r,f,w,d,a,t,z,d,f,s],[q,w,e,r,t,z]).
[a, f, d, s, q, w, e, r, t, z]

Yes
```

Semantik der PL

Eine **Struktur** ist ein Paar $\mathcal{A}=(\mathcal{U}_\mathcal{A},\mathcal{I}_\mathcal{A})$.
$\mathcal{U}_\mathcal{A}$ („Universum“) umfasst Objekte, Funktionen, Relationen.
$\mathcal{I}_\mathcal{A}$ („Interpretation“) ist eine Abbildung, die abbildet:

- Prädikatensymbole P_i^k auf k-stellige Relation aus $\mathcal{U}_\mathcal{A}$;
- Funktionssymbole f_i^k auf k-stellige Funktion aus $\mathcal{U}_\mathcal{A}$;
- Variablen auf Objekte aus $\mathcal{U}_\mathcal{A}$.

Abkürzende Schreibweisen: $x_\mathcal{A}:=\mathcal{I}_\mathcal{A}(x)$, ebenso für f, P.

$\mathcal{A}(t)$ (Wert eines Terms t in $\mathcal{A}$):

- Falls t Variable x ist: $\mathcal{A}(t)=x_\mathcal{A}$;
- Falls t Funktion $f_i^k(t_1,\ldots,t_k)$ ist: $\mathcal{A}(t) = f_\mathcal{A}(\mathcal{A}(t_1),\ldots,\mathcal{A}(t_k))$

Semantik der PL: Wert einer Formel in $\mathcal{A}$

- $\mathcal{A}(P_i^k(t_1,\ldots,t_k)) = 1$, falls $(\mathcal{A}(t_1),\ldots,\mathcal{A}(t_k)) \in P_{\mathcal{A}}$; sonst 0
- $\mathcal{A}(\neg\alpha) = 1$, falls $\mathcal{A}(\alpha)=0$; sonst 0
- [entsprechend für $\wedge$, $\vee$]
- $\mathcal{A}(\forall x.\alpha) = 1$, falls für alle $d \in \mathcal{U}_{\mathcal{A}}$ gilt $\mathcal{A}_{[x/d]}(\alpha)=1$; sonst 0
- $\mathcal{A}(\exists x.\alpha) = 1$, falls es ein $d \in \mathcal{U}_{\mathcal{A}}$ gibt mit $\mathcal{A}_{[x/d]}(\alpha)=1$; sonst 0

$\mathcal{A}_{[x/d]}$ ist die Struktur, die mit $\mathcal{A}$ identisch ist bis auf den Wert v. x. Der Wert von x in $\mathcal{A}_{[x/d]}$ ist d.

Falls $\mathcal{A}(\alpha)=1$, heißt $\mathcal{A}$ Modell von α.

α heißt allgemeingültig, falls alle „syntaktisch passenden“ Strukturen Modelle von α sind.

Unentscheidbarkeit

Satz von Church

Das Gültigkeitsproblem der Prädikatenlogik ist unentscheidbar.

Beweis s. Schöning Kap. 2.3 (Rückführung auf Postsches Korrespondenzproblem)

Satz

Das Erfüllbarkeitsproblem der Prädikatenlogik ist unentscheidbar.

Beweis s. Schöning Kap. 2.3 (Rückführung auf Satz v. Church)

Satz

Erfüllbarkeit und Gültigkeit sind semi-entscheidbar:
Es gibt Kalküle, die genau alle gültigen Formeln herleiten.

Beweis s.u.: Wir werden solche Verfahren kennen lernen

Bemerkung: Es gibt entscheidbare Untermengen der PL (z.B. AL)

Äquivalenzen

… aus der AL gelten entsprechend weiter (s. Folie 121)! Zusätzlich:

$\neg \forall x.\alpha \equiv \exists x.\neg\alpha$	deMorgansche Regel
$\neg \exists x.\alpha \equiv \forall x.\neg\alpha$	deMorgansche Regel
$(\forall/\exists x.\ \alpha \diamond \beta) \equiv \forall/\exists x.(\alpha \diamond \beta)$	Skopuserweiterung falls x nicht frei in β; $\diamond \in \{\wedge, \vee\}$
$(\forall x.\alpha \wedge \forall x.\beta) \equiv \forall x.\ (\alpha \wedge \beta)$ $(\exists x.\alpha \vee \exists x.\beta) \equiv \exists x.\ (\alpha \vee \beta)$	
$\forall x.\forall y.\ \alpha \equiv \forall y.\forall x.\ \alpha$ $\exists x.\exists y.\ \alpha \equiv \exists y.\exists x.\ \alpha$	

Skolemform

Sei $\alpha = Q_1 x_1. \ldots Q_n x_n.\alpha'$ Formel in **Pränex-Normalform**, d.h. $Q_1,\ldots,Q_n$ sind Quantoren und α' ist quantorenfrei. Seien $Q_1,\ldots,Q_i$ ($i \in \{0,\ldots,n\}$) Allquantoren, sei also Q_{i+1} der erste auftretende Existenzquantor. Sei f^i eine neue, in α' nicht vorkommende i-stellige Funktion, sog. **Skolem-Funktion**.

Die Ersetzung von x_{i+1} überall in α' durch $f^i(x_1,\ldots, x_i)$ und Streichung des Quantors $Q_{i+1}x_{i+1}$ heißt **∃-Elimination** oder **Skolemisierung**.

Eine Formel in Pränex-Normalform, in der *alle* ∃-Quantoren eliminiert sind, heißt in **Skolemform**.

Skolemisierung

Beispiel

Skolemisierung von $\forall x.\ \exists y.\ \exists t.\ \text{loves}(x,y,t)$

für Skolem-Funktionen $f(.)$, $g(.)$: $\forall x.\ \text{loves}(x,f(x),g(x))$

Intuitive Interpretation

$f(.)$ bezeichnet den/die Geliebte/n, $g(.)$ den „goldenen Moment“

Ist Skolemisierung eine Äquivalenzumformung?

Satz

Überführen einer Formel in Skolemform ist inkonsistenzerhaltend.

Beweis: Betrachtung der Modelle

Umformung prädikatenlogischer Formeln in KNF

1. Benenne gebundene Variablen eindeutig
2. Löse $\Leftrightarrow$ und $\Rightarrow$ auf (s. Folie 116)
3. Bringe alle Negationszeichen direkt vor die Atomformeln, löse dabei doppelte Negation immer auf
   (deMorgansche Regeln, auch für Quantoren, Folie 174)
4. Schiebe alle Quantoren nach links
   (Äquivalenzen Folie 174)
5. Überführe den Teil „hinter den Quantoren“ in KNF wie in AL, falls noch erforderlich

Beispiel: Forme um: $\forall x.\forall y.[\exists z.(P(x,z) \wedge P(y,z)) \Rightarrow \exists z.Q(x,y,z)]$

an der Tafel

Tafelbeispiel

$$\forall x.\forall y.\Big[\exists z.\big(P(x,z)\wedge P(y,z)\big)\Rightarrow \exists z.Q(x,y,z)\Big]$$

$$\begin{aligned}
&\equiv \forall x.\forall y.\Big[\exists u.\big(P(x,u)\wedge P(y,u)\big)\Rightarrow \exists v.Q(x,y,v)\Big]\\
&\equiv \forall x.\forall y.\Big[\neg\exists u.\big(P(x,u)\wedge P(y,u)\big)\vee \exists v.Q(x,y,v)\Big]\\
&\equiv \forall x.\forall y.\Big[\forall u.\neg\big(P(x,u)\wedge P(y,u)\big)\vee \exists v.Q(x,y,v)\Big]\\
&\equiv \forall x.\forall y.\Big[\forall u.\big(\neg P(x,u)\vee\neg P(y,u)\big)\vee \exists v.Q(x,y,v)\Big]\\
&\equiv \forall x.\forall y.\forall u.\Big[\neg P(x,u)\vee\neg P(y,u)\vee \exists v.Q(x,y,v)\Big]\\
&\equiv \forall x.\forall y.\forall u.\exists v.\Big[\neg P(x,u)\vee\neg P(y,u)\vee Q(x,y,v)\Big]\\
&\equiv \forall x.\forall y.\forall u.\Big[\neg P(x,u)\vee\neg P(y,u)\vee Q(x,y,f(x,y,u))\Big]
\end{aligned}$$

Klauseln in der Prädikatenlogik

Analog zur Aussagenlogik definieren wir

Ein **Literal** ist eine negierte oder nicht negierte *Atomformel*.
Eine **Klausel** ist eine Disjunktion von Literalen

Umformen einer Formel in KNF und Skolemform in Klauselform:

- Lass alle (All-)Quantoren weg;
- interpretiere alle Variablen als allquantifiziert;
- stelle Klausel als Menge von Literalen dar;
- stelle KNF-Formel als Menge von Klauseln dar.

Beispiel

$$\forall x.\forall y.\forall u.[\neg P(x,u) \vee \neg P(y,z) \vee Q(x,y,f(x,y,u))]$$

➥ $$\{\{\neg P(x,u), \neg P(y,z), Q(x,y,f(x,y,u))\}\}$$

Definitionen aus der Herbrand-Theorie

Das **Herbrand-Universum** H_S einer Klauselmenge S ist die kleinste abgeschlossene Menge aller Grundterme, die folgendermaßen gebildet werden:

1. Alle in S vorkommenden Konstanten sind in H_S.
   Kommt keine Konstante in S vor, ist die Konstante a in H_S.
2. Für jede Funktion f_i^k in S und Terme $t_1,\ldots,t_k$ ist $f_i^k(t_1,\ldots,t_k) \in H_S$.

Die **Herbrand-Basis** $H_S(S)$ einer Klauselmenge S ist die Menge der Grundklauseln, die aus S durch konsistentes Einsetzen aller Elemente aus H_S für alle Variablen in allen Klauseln entstehen.

Der Satz von Herbrand

Details s. Schöning Kap.2.4

Satz von Herbrand (Version für Klauselform)

Eine Menge S von Klauseln ist inkonsistent, gdw. eine endliche Teilmenge der Herbrand-Basis $H_S(S)$ existiert, die inkonsistent ist.

Beweis: Betrachtung der „Herbrand-Modelle“ von S (hier nicht definiert):
Satz von Gödel/Herbrand/Skolem in Kombination mit Endlichkeitssatz für AL.

Vor Einführung der Resolution wurden Algorithmen untersucht, die diesen Satz „naiv“ implementierten:

Erzeuge systematisch die Herbrand-Basis;
prüfe jede endliche Teilmenge aussagenlogisch auf Inkonsistenz

Schritt 1 Richtung Resolution: Substituieren

Wegen impliziter Allquantifizierung der Variablen gilt:

- $P(x)$, $\neg P(y)$ widersprüchlich;
- $P(x)$ $\neg P(f(a))$ widersprüchlich;

aber *nicht*

- $P(x)$, $\neg P(f(x))$!

Für vollständige Inferenzverfahren müssen wir semantisch gleiche Literale auch textuell gleich machen, unifizieren können!

Eine **Substitution** ist eine Folge von Ersetzungen x/t einer Variablen x durch einen Term t, wobei x in t nicht vorkommt.

Bemerkung: Durch Anwendung einzelner Ersetzungen auf den ersetzenden Term früherer Ersetzungen erzeuge Reihenfolgeunabhängigkeit der Ersetzungen

➥ Substitutionen werden *Mengen* von x/t-Paaren.