Prädikatenlogische Entscheidbarkeitsprobleme

Erfüllbarkeitsproblem:

Gegeben: prädikatenlogischer Ausdruck A über einer Signatur $\mathcal{S}$
Frage: Ist A erfüllbar ?

Gültigkeitsproblem:

Gegeben: prädikatenlogischer Ausdruck A über einer Signatur $\mathcal{S}$
Frage: Ist A allgemeingültig ?

Unerfüllbarkeitsproblem:

Gegeben: prädikatenlogischer Ausdruck A über einer Signatur $\mathcal{S}$
Frage: Ist A unerfüllbar ?

Algorithmus und Entscheidbarkeit

Ein *Algorithmus* überführt in endlicher Zeit die Eingabedaten in eine Antwort "ja" oder "nein" und besteht aus einer Folge von Anweisungen mit folgenden Eigenschaften:

- es gibt eine eindeutig festgelegte Anweisung, die als erste auszuführen ist,

- nach Abarbeitung einer Anweisung gibt es eine eindeutig festgelegte Anweisung, die als nächste abzuarbeiten ist, oder die Abarbeitung des Algorithmus ist beendet.

Ein Problem heißt *entscheidbar*, wenn es einen Algorithmus zu seiner Lösung gibt, d.h. einen Algorithmus, der auf die Frage die korrekte Antwort "ja" oder "nein" gibt. Falls kein Algorithmus zur Lösung existiert, heißt das Problem *unentscheidbar*.

Unentscheidbarkeit in der Prädikatenlogik

Satz:
Das Gültigkeitsproblem der Prädikatenlogik ist unentscheidbar, d.h. es gibt keinen Algorithmus, der für einen beliebigen prädikatenlogischen Ausdruck A feststellt, ob A eine Tautologie ist.

Satz:
Das Unerfüllbarkeitsproblem der Prädikatenlogik ist unentscheidbar, d.h. es gibt keinen Algorithmus, der für einen beliebigen prädikatenlogischen Ausdruck A feststellt, ob A unerfüllbar ist.

Satz:
Das Erfüllbarkeitsproblem der Prädikatenlogik ist unentscheidbar, d.h. es gibt keinen Algorithmus, der für einen beliebigen prädikatenlogischen Ausdruck A feststellt, ob A erfüllbar ist.

Herbrand-Theorie I

Definition:
Für einen prädikatenlogischen Ausdruck A in bereinigter Skolemform definieren das Herbrand-Universum $H(A)$ von A induktiv wie folgt:

- Alle in A vorkommenden Konstanten sind in $H(A)$. Falls A keine Konstante enthält, so ist $a \in H(A)$ (wobei a ein Symbol ist, das in A nicht vorkommt).

- Sind $t_1, t_2, \ldots, t_k$ in $H(A)$ und ist f ein k-stelliges Funktionssymbol in A, so ist $f(t_1, t_2, \ldots, t_k) \in H(A)$.

Herbrand-Theorie II

Definition: Für einen Ausdruck $A = \forall x_1 \forall x_2 \ldots \forall x_n A'$ in bereinigter Skolemform definieren wir die Herbrand-Erweiterung $E(A)$ als die Menge

$$E(A) = \{sub(sub(\ldots(sub(sub(A', x_n, t_n), x_{n-1}, t_{n-1})\ldots), x_2, t_2), x_1, t_1) \mid t_1, t_2, \ldots, t_n \in H(A)\}$$

und setzen

$$E'(A) = \{sub(\ldots(sub(sub(B, x_n, t_n), x_{n-1}, t_{n-1})\ldots), x_1, t_1) \mid B \in B(A), t_1, t_2, \ldots, t_n \in H(A)\} \ .$$

Satz: Ein Ausdruck A in bereinigter Skolemform ist genau dann erfüllbar, wenn die Menge $E(A)$ im aussagenlogischen Sinn erfüllbar ist (d.h., wenn es eine Belegung α von $E'(A)$ derart gibt, dass $w_\alpha(B) = 1$ für alle $B \in E(A)$ gilt).

Semi-Algorithmus (von GILMORE) für das Unerfüllbarkeitsproblem

Wir nennen ein Verfahren zur Entscheidung einer Eigenschaft E einen Semi-Algorithmus, wenn es

— auf einer Eingabe X mit Eigenschaft E nach einer endlichen Anzahl von Schritten antwortet, dass X die Eigenschaft E hat und

— auf einer Eingabe, die die Eigenschaft E nicht hat, keine Antwort gibt.

Eingabe: prädikatenlogischer Ausdruck A in bereinigter Skolemform, Aufzählung von $E(A) = \{A_1, A_2, A_3, \ldots\}$

$n = 1$; $F = A_1$;
`while` (F ist erfüllbar) $\{$ $n = n + 1$; $F = F \wedge A_n$;$\}$
Gib "A ist unerfüllbar" aus (und stoppe);

Grundresolutionsalgorithmus zur Entscheidung der Unerfüllbarkeit eines prädikatenlogischen Ausdrucks

Eingabe: präd.log. Ausdruck $A = \forall x_1 \ldots \forall x_k A'$ in bereinigter Skolemform
A' in konjunktiver Normalform
Aufzählung von $E(A) = \{A_1, A_2, A_3, \ldots\}$
für $n \geq 1$ sei K_n die Klauselmenge zu A_n

$n = 1$; $M = \{K_1\}$; $M = res^*(M)$;
`while` $(\emptyset \notin M)$ { $n = n + 1$; $M = M \cup \{K_n\}$; $M = res^*(M)$; }
Gib "A ist unerfüllbar" aus (und stoppe);

Unifikator

Definition:
i) Eine Substitution s heißt Unifikator der Menge $\mathcal{L} = \{L_1, L_2, \ldots L_r\}$ von Literalen, falls $s(L_1) = s(L_2) = \ldots = s(L_r)$ gelten.

ii) Eine Substitution s heißt allgemeinster Unifikator von $\mathcal{L}$, falls für jeden Unifikator s' von $\mathcal{L}$ eine Substitution s'' mit $s' = s \circ s''$ existiert.

Satz:
Jede unifizierbare Menge von Literalen besitzt einen allgemeinsten Unifikator.

Algorithmus für den allgemeinsten Unifikator

Eingabe: nichtleere Menge $\mathcal{L} = \{L_1, L_2, \ldots, L_r\}$ von Literalen

$s = id$;

`while` ($s(L_i) \neq s(L_j)$ für gewisse $1 \leq i < j \leq k$)

{ Durchsuche die Literale von $s(L_i)$ und $s(L_j)$ von links nach rechts,
bis erste Position a gefunden ist, an der sich mindestens
zwei Literale unterscheiden;
`if` (keines der Zeichen ist eine Variable)
Stoppe mit "$\mathcal{L}$ ist nicht unifizierbar";
`else` { x = Variable in a; t = Term, der in a beginnt;
`if` (x kommt in t vor)
Stoppe mit "$\mathcal{L}$ ist nicht unifizierbar";
`else` $s = s \circ [x/t]$;
}
}

Gib s als allgemeinsten Unifikator aus

Unifikationsalgorithmus – Beispiel

Literale: $L_1 = \neg P(f(z, g(a, y)), h(z))$
$L_2 = \neg P(f(f(u, v), w), h(f(a, b)))$

Unterschied an der sechsten Position: Substitution $s_1 = [z/f(u, v)]$
$s_1(L_1) = \neg P(f(f(u, v), g(a, y)), h(f(u, v)))$
$s_1(L_2) = \neg P(f(f(u, v), w), h(f(a, b)))$

Unterschied an der elften Position: Substitution $s_2 = [w/g(a, y)]$
$s_2(s_1(L_1)) = \neg P(f(f(u, v), g(a, y)), h(f(u, v)))$
$s_2(s_1(L_2)) = \neg P(f(f(u, v), g(a, y)), h(f(a, b)))$

Unifikationsalgorithmus – Beispiel – Fortsetzung

Literale: $s_2(s_1(L_1)) = \neg P(f(f(u,v),g(a,y)),h(f(u,v)))$
$s_2(s_1(L_2)) = \neg P(f(f(u,v),g(a,y)),h(f(a,b)))$

Unterschied bei sechstletzten Buchstaben: Substitution $s_3 = [u/a]$

$s_3(s_2(s_1(L_1))) = \neg P(f(f(a,v),g(a,y)),h(f(a,v)))$
$s_3(s_2(s_1(L_2))) = \neg P(f(f(a,v),g(a,y)),h(f(a,b)))$

Unterschied bei viertletzten Buchstaben: Substitution $s_4 = [v/b]$

$s_4(s_3(s_2(s_1(L_1)))) = \neg P(f(f(a,b),g(a,y)),h(f(a,b)))$
$s_4(s_3(s_2(s_1(L_2)))) = \neg P(f(f(a,b),g(a,y)),h(f(a,b)))$

allgemeinsten Unifikator von L_1 und L_2:

$s_1 \circ s_2 \circ s_3 \circ s_4 = [z/f(u,v)] \circ [w/g(a,y)] \circ [u/a] \circ [v/b]$

Prädikatenlogische Resolution – Resolvente

Definition:
Es seien $\mathcal{K}_1$, $\mathcal{K}_2$ und $\mathcal{R}$ Mengen von prädikatenlogischen Literalen. Dann heißt $\mathcal{R}$ prädikatenlogische Resolvente von $\mathcal{K}_1$ und $\mathcal{K}_2$, falls folgende Bedingungen erfüllt sind:

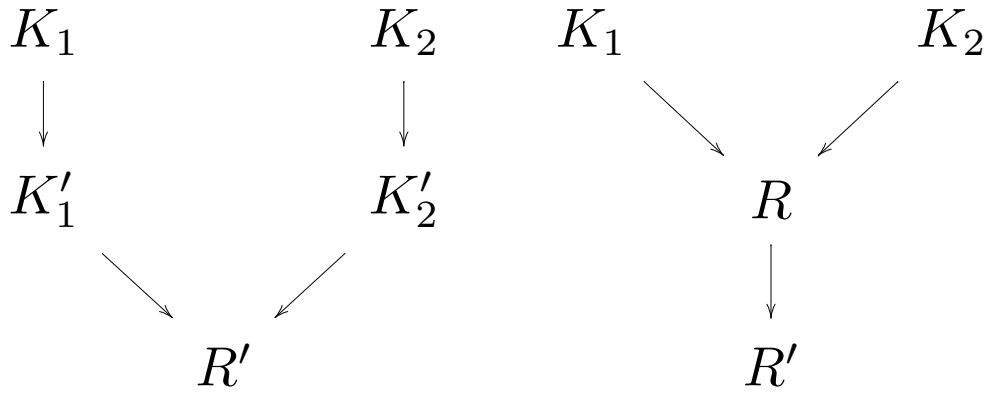
- Es gibt Substitutionen s_1 und s_2, die nur Variablenumbenennugen sind, so dass $s_1(\mathcal{K}_1)$ und $s_2(\mathcal{K}_2)$ keine gemeinsamen Variablen haben.

- Es gibt Literale $L_1, \ldots, L_m \in s_1(\mathcal{K}_1)$, $m \geq 1$, und $L'_1, \ldots, L'_n \in s_2(\mathcal{K}_2)$, $n \geq 1$, so dass die Menge $\mathcal{L} = \{\neg L_1, \neg L_2, \ldots, \neg L_m, L'_1, L'_2, \ldots, L'_n\}$ unifizierbar ist.
  s sei der allgemeinste Unifikator von $\mathcal{L}$.

- Es gilt $\mathcal{R} = s((s_1(\mathcal{K}_1) \setminus \{L_1, L_2, \ldots, L_m\}) \cup (s_2(\mathcal{K}_2) \setminus \{L'_1, L'_2, \ldots, L'_n\}))$

Prädikatenlogische Resolution – Lifting-Lemma

Lemma:
Es seien K_1 und K_2 zwei prädikatenlogische Klauseln und K_1' und K_2' zugehörige (beliebige) Grundinstanzen. Ferner sei R' eine (aussagenlogische) Resolvente von K_1' und K_2'. Dann gibt es eine prädikatenlogische Resolvente R von K_1 und K_2 so, dass R' Grundinstanz von R ist.



Prädikatenlogische Resolution – Resolutionshülle

Definition: Für eine Menge $\mathcal{F}$ von Mengen von Literalen setzen wir

$$\begin{aligned} Res(\mathcal{F}) &= \mathcal{F} \cup \{\mathcal{R} \mid \mathcal{R} \text{ ist Resolvente gewisser } \mathcal{K} \in \mathcal{F} \text{ und } \mathcal{K}' \in \mathcal{F}\}, \\ Res^0(\mathcal{F}) &= \mathcal{F}, \\ Res^n(\mathcal{F}) &= Res(Res^{n-1}(\mathcal{F})) \text{ für } n \geq 1 \text{ und} \\ Res^*(\mathcal{F}) &= \bigcup_{n \geq 0} Res^n(\mathcal{F}). \end{aligned}$$

Satz: Ein prädikatenlogischer Ausdruck $A = \forall x_1 \forall x_2 \ldots \forall x_n A'$ in bereinigter Skolemform, bei dem A' in konjunktiver Normalform vorliegt, ist A genau dann unerfüllbar, wenn die leere Menge in $Res^*(A)$ liegt.

Logik-Programme

Definition:

Eine Tatsachenklausel ist eine einelementige positive Klausel, d.h. sie hat die Form $\{P\}$.

Eine Prozedurklausel ist eine Klausel der Form $\{P, \neg Q_1, \neg Q_2, \ldots, \neg Q_k\}$ mit $k \geq 1$.
P heißt Prozedurkopf, und $Q_1, Q_2, \ldots, Q_k$ bilden den Prozedurkörper.

Ein Logik-Programm ist eine endliche Menge von Tatsachen- und Prozedurklauseln.

Eine Zielklausel ist eine Klausel der Form $\{\neg Q_1, \neg Q_2, \ldots, \neg Q_k\}$ mit $k \geq 1$.

Konfigurationen und ihre Übergänge

Definition: Es sei F ein Logik-Programm.

i) Eine Konfiguration ist ein Paar (G, sub), wobei G eine Zielklausel und sub eine Substitution ist.

ii) Wir sagen, dass die Konfiguration (G, sub) bez. F in die Konfiguration (G', sub') überführt wird (und schreiben $(G, sub) \vdash_F (G', sub')$), falls folgende Bedingungen erfüllt sind:

— $G = \{\neg Q_1, \neg Q_2, \ldots, \neg Q_k\}$

— es gibt in F eine Klausel $K = \{P, \neg A_1, \neg A_2, \ldots, \neg A_n\}$, $n \geq 0$, und ein i, $1 \leq i \leq n$, so dass B (nach einigen Umbenennungen) mit Q_i unifizierbar ist,

— $G' = s(\{\neg Q_1, \ldots, \neg Q_{i-1}, \neg A_1, \ldots, \neg A_n, \neg Q_{i+1}, \ldots, \neg Q_k\}$, wobei s der allgemeinste Unifikator von B und Q_i ist,

— $sub' = sub \circ s$.

Berechnungen

Definition:

Es seien F ein Logik-Programm und $G = \{\neg Q_1, \ldots, \neg Q_k\}$ eine Zielklausel.

i) Eine Berechnung von F bei Eingabe von G ist eine Folge der Form

$$(G, id) \vdash_F (G_1, sub_1) \vdash (G_2, sub_2) \vdash_F \ldots \vdash_F (G_n, sub_n) \vdash_F \ldots \ .$$

ii) Falls eine Rechnung endlich ist und für das letzte Glied (G_n, sub) der Folge $G_n = \emptyset$ gilt, so heißt die Berechnung erfolgreich und $sub(Q_1 \wedge Q_2 \wedge \ldots \wedge Q_k)$ ist das Ergebnis der Rechnung.

n ist die Länge der Berechnung.

Korrektheit und Vollständigkeit

Satz:
Seien F ein Logik-Programm und G eine Zielklausel.
Falls es eine erfolgreiche Rechnung von F bei Eingabe von G gibt, so ist jede Grundinstanz des Rechenergebnisses eine Folgerung von F.

Satz:
Seien F ein Logik-Programm und $G = \{\neg Q_1, \ldots, \neg Q_k\}$ eine Zielklausel.
Falls jede Grundinstanz von $(Q_1 \wedge \ldots \wedge Q_k)$eine Folgerung von F ist, so gibt es eine erfolgreiche Rechnung von F bei Eingabe von G mit dem Ergebnis $sub(Q_1 \wedge Q_2 \wedge \ldots Q_k)$, und für jede Grundinstanz $sub'(Q_1 \wedge Q_2 \wedge \ldots Q_k)$ gibt es eine Substitution s mit

$$sub'(Q_1 \wedge Q_2 \wedge \ldots Q_k) = s(sub(Q_1 \wedge Q_2 \wedge \ldots Q_k)) \,.$$

Kanonische Berechnungen

Definition:
Seien F ein Logik-Programm und G eine Zielklausel.
Eine Rechnung von F bei Eingabe von G heißt kanonisch, falls in jeder Konfigurationsüberführung $(G', sub') \vdash_F (G'', sub'')$ nach dem ersten (d.h. dem am weitesten links stehenden) Literal von G' resolviert wird.

Satz:
Seien F ein Logik-Programm und G eine Zielklausel.
Falls es eine erfolgreiche Rechnung $\mathcal{R}$ von F bei Eingabe von G gibt, so gibt es auch eine erfolgreiche kanonische Rechnung $\mathcal{R}'$ von F bei Eingabe von G, so dass $\mathcal{R}$ und $\mathcal{R}'$ die gleiche Länge haben und das gleiche Ergebnis liefern.

Vollständigkeit von Strategien

Definition: Eine Strategie heißt vollständig, wenn es für jedes Logik-Programm F und jede Zielklausel G, für die es eine erfolgreiche Berechnung von F bei Eingabe von G gibt, auch eine erfolgreiche Berechnung von F bei Eingabe von G mittels der Strategie gibt.

Satz: Die Breitensuche ist eine vollständige Strategie.

Satz: Die Tiefensuche ist keine vollständige Strategie.