TECHNISCHE UNIVERSITÄT DRESDEN

THEORETISCHE INFORMATIK UND LOGIK

16. Vorlesung: Logisches Schließen (2)

Markus Krötzsch

Professur Wissensbasierte Systeme

TU Dresden, 17. Juni 2024

Schließen ist schwer

Erinnerung: F ist logische Konsequenz von G, wenn alle Modelle von G auch Modelle von F sind.

- Es ist nicht offensichtlich, wie man das überprüfen sollte, denn es gibt unendliche viele Modelle
- Ebenso schwer erscheinen die gleichwertigen Probleme der Erfüllbarkeit und Allgemeingültigkeit

Intuition: prädikatenlogisches Schließen ist unentscheidbar

Wie kann man das beweisen?

Durch Reduktion eines bekannten unentscheidbaren Problems, z.B.

- Halteproblem
- Postsches Korrespondenzproblem
- Äquivalenz kontextfreier Sprachen
- ...

Unentscheidbarkeit (1)

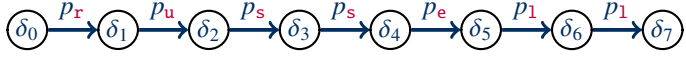
Satz: Logisches Schließen (Erfüllbarkeit, Allgemeingültigkeit, logische Konsequenz) in der Prädikatenlogik ist unentscheidbar.

Beweis: Durch Reduktion vom CFG-Schnittproblem:

Gegeben: Kontextfreie Grammatiken G_1 und G_2
Frage: Gibt es ein Wort $w \in \mathbf{L}(G_1) \cap \mathbf{L}(G_2)$?

Idee: Wir kodieren Wörter in der Prädikatenlogik als Ketten von binären Relationen.

Zum Beispiel würde das Wort `russell` in einer Modellstruktur $\mathcal{I}$ wie folgt aussehen:



Diese Skizze soll bedeuten, dass z.B. $\langle \delta_2, \delta_3 \rangle, \langle \delta_3, \delta_4 \rangle \in p_{\mathtt{s}}^{\mathcal{I}}$. Wir verwenden ein Prädikatensymbol $p_{\mathtt{a}}$ für jedes Alphabetssymbol `a`.

Unentscheidbarkeit (2)

Satz: Logisches Schließen (Erfüllbarkeit, Allgemeingültigkeit, logische Konsequenz) in der Prädikatenlogik ist unentscheidbar.

Beweis (Fortsetzung): Zusätzlich verwenden wir binäre Prädikatensymbole p_A für jedes Nichtterminalsymbol A.

Die Kodierung von Grammatiken ist nun direkt möglich:

- Wir nehmen o.B.d.A. an, dass G_1 und G_2 keine Nichtterminale gemeinsam haben.
- Eine Produktionsregel $A \to \sigma_1 \cdots \sigma_n$ kodieren wir als Formel:

$$\forall x_0, \ldots, x_n.((p_{\sigma_1}(x_0, x_1) \wedge \ldots \wedge p_{\sigma_n}(x_{n-1}, x_n)) \to p_A(x_0, x_n))$$

- Idee: die Formel erkennt, ob eine gegebene Kette aus Terminalen und Nichtterminalen aus einem anderen Nichtterminal entstehen kann

Unentscheidbarkeit (3)

Satz: Logisches Schließen (Erfüllbarkeit, Allgemeingültigkeit, logische Konsequenz) in der Prädikatenlogik ist unentscheidbar.

Beweis (Fortsetzung): Seien S_1 und S_2 die Startsymbole von G_1 und G_2. Dann wollen wir das Schnittproblem kodieren, indem wir fragen, ob die folgende Formel folgt:

$$\exists x, y.(p_{S_1}(x, y) \wedge p_{S_2}(x, y))$$

Was fehlt?

- Die Grammatik-Formeln können erkennen, ob eine Zeichenkette aus S_1 oder S_2 abgeleitet werden kann
- Jede Interpretation, welche die Kodierung eines Wortes $w \in \mathbf{L}(G_1) \cap \mathbf{L}(G_2)$ enthält, muss daher auch die Formel $\exists x, y.(p_{S_1}(x, y) \wedge p_{S_2}(x, y))$ erfüllen
- Aber: Es kann auch Interpretationen geben, welche keine Kodierung von w enthalten

$\rightsquigarrow$ Alle möglichen Wörter müssten kodiert vorkommen ...

Unentscheidbarkeit (4)

Satz: Logisches Schließen (Erfüllbarkeit, Allgemeingültigkeit, logische Konsequenz) in der Prädikatenlogik ist unentscheidbar.

Beweis (Fortsetzung): Alle möglichen Wörter müssten kodiert vorkommen … Dazu fügen wir noch folgende Sätze hinzu:

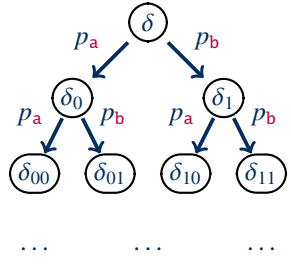
$$\forall x.\exists y.p_{\mathsf{a}}(x,y) \qquad \text{für jedes Terminalsymbol } \mathsf{a}$$

- Jedes Modell dieser Theorie muss Kodierungen aller Wörter enthalten (aber eventuell als zyklische oder überlappende Pfade, z.B. in einem Modell mit nur einem Element, welches in jeder möglichen Relation zu sich selbst steht)
- Gibt es ein Wort $w \in \mathbf{L}(G_1) \cap \mathbf{L}(G_2)$, dann muss auch dieses Wort kodiert vorkommen: es folgt $\exists x, y.(p_{\mathsf{S}_1}(x,y) \wedge p_{\mathsf{S}_2}(x,y))$
- Folgt $\exists x, y.(p_{\mathsf{S}_1}(x,y) \wedge p_{\mathsf{S}_2}(x,y))$, erfüllen alle Modelle diesen Satz, speziell auch das Modell, welches man erhält, indem man einen unendlichen Baum aller Wörter aufbaut

Unentscheidbarkeit (5)

Satz: Logisches Schließen (Erfüllbarkeit, Allgemeingültigkeit, logische Konsequenz) in der Prädikatenlogik ist unentscheidbar.

Beweis (Fortsetzung): Skizze des „kanonischen“ baumförmigen Modells, welches keine Zyklen oder parallelen Relationen enthält, über Alphabet $\{\mathsf{a}, \mathsf{b}\}$ (ohne eventuelle Kanten für Nichtterminale):



Falls „sogar“ dieses Modell $\exists x, y.(p_{\mathsf{S}_1}(x, y) \wedge p_{\mathsf{S}_2}(x, y))$ erfüllt, dann muss es ein Wort $w \in \mathbf{L}(G_1) \cap \mathbf{L}(G_2)$ geben.

Unentscheidbarkeit (6)

Satz: Logisches Schließen (Erfüllbarkeit, Allgemeingültigkeit, logische Konsequenz) in der Prädikatenlogik ist unentscheidbar.

Beweis (Zusammenfassung): Wir konstruieren aus G_1 und G_2 eine logische Theorie $\mathcal{T}$ mit den folgenden Sätzen:

- für jede Produktionsregel $\mathsf{A} \to \sigma_1 \cdots \sigma_n$ von G_1 oder G_2:

$$\forall x_0, \ldots, x_n.((p_{\sigma_1}(x_0, x_1) \wedge \ldots \wedge p_{\sigma_n}(x_{n-1}, x_n)) \to p_{\mathsf{A}}(x_0, x_n))$$

- Für jedes Terminalsymbol a:

$$\forall x.\exists y.p_{\mathsf{a}}(x, y)$$

Dann gilt $\mathcal{T} \models \exists x, y.(p_{\mathsf{S}_1}(x, y) \wedge p_{\mathsf{S}_2}(x, y))$ genau dann wenn es ein Wort $w \in \mathbf{L}(G_1) \cap \mathbf{L}(G_2)$ gibt.

Die Unentscheidbarkeit von Erfüllbarkeit und Allgemeingültigkeit folgt, weil man logische Konsequenz auf diese Probleme reduzieren kann. □

Einfache Folgerungen

Offensichtlich wird das Schließen nicht einfacher, wenn man auch noch Gleichheit erlaubt:

Korollar: Logisches Schließen (Erfüllbarkeit, Allgemeingültigkeit, logische Konsequenz) in der Prädikatenlogik mit Gleichheit ist unentscheidbar.

Umgekehrt kann man aus dem Beweis noch stärkere Ergebnisse folgern, z.B.:

Korollar: Logisches Schließen (Erfüllbarkeit, Allgemeingültigkeit, logische Konsequenz) in der Prädikatenlogik ist unentscheidbar, selbst dann, wenn nur binäre Prädikatssymbole verwendet werden.

Unentscheidbarkeit hat viele Beweise

Unser Unentscheidbarkeitsbeweis ist nicht der Einzige ...

- Mit Prädikatenlogik kann man viele mathematische Definitionen direkt ausdrücken
- Viele Probleme lassen sich dadurch ziemlich einfach kodieren
- Der erste Schritt ist am wichtigsten:
  Wie wollen wir das Problem in logischen Interpretationen mit Relationen darstellen?
- Sind die Prädikatensymbole einmal festgelegt, dann muss man nur noch die Problemdefinition in diese Kodierung übersetzen.

Beispiel: Wir könnten das Halteproblem von TMs direkt kodieren, aber dabei benötigt man viel mehr Prädikate. Es ist nicht schwer, Aussagen zu kodieren wie z.B. „Falls es einen Zeitpunkt t gibt, an dem die TM in Zustand q an Position y das Zeichen a liest, dann folgt auf t ein Zeitpunkt t', an dem die TM in Zustand q' ist.“ Der Beweis ist leicht, sofern man keine relevante Aussage vergisst.

Gödel

Zwischenstand

Wir haben bisher zwei Dinge erkannt:

(1) Prädikatenlogik ist sehr ausdrucksstark: mit ihr können wir z.B. das mathematische Konzept der Gleichheit axiomatisieren oder Probleme über formale Sprachen beschreiben

(2) Prädikatenlogisches Schließen ist unentscheidbar: man kann Probleme definieren, die nicht mehr algorithmisch lösbar sind

Eignet sich Prädikatenlogik also als universelle Beschreibungssprache für die Mathematik (und darüber hinaus)?

Dazu hat Kurt Gödel (1906–1978) einiges zu sagen ...

Gödels Einsichten

Gödel zeigte einige wesentliche Dinge:

(1) Gödelscher Vollständigkeitssatz (1930):
„Es gibt ein konsistentes Verfahren, das alle Konsequenzen einer prädikatenlogischen Theorie effektiv beweisen kann.“
  - Alle wahren Sätze können endlich bewiesen werden
  - Prädikatenlogisches Schließen ist semi-entscheidbar

(2) 1. Gödelscher Unvollständigkeitssatz (1931):
„Es gibt kein konsistentes Verfahren, das alle Konsequenzen der elementaren Arithmetik effektiv beweisen kann.“
  - Für jedes Verfahren gibt es allgemeingültige Sätze über elementare arithmetische Zusammenhänge, die nicht bewiesen werden können
  - Die Wahrheit elementarer arithmetischer Zusammenhänge ist nicht semi-entscheidbar

(3) 2. Gödelscher Unvollständigkeitssatz (1931): später

Mehr dazu in späteren Vorlesungen . . .

Algorithmen zum logischen Schließen

Wie können wir logisch Schließen?

Gödel betrachtete ein konkretes Verfahren zum logischen Schließen, basierend auf:

- einer Menge an vorgegebenen Tautologien
- einer Menge von Regeln, mit denen man aus bereits bekannten Tautologien neue ableiten kann

Diese Art von Verfahren war seit der Antike bekannt.

Gödel zeigte, dass dieses Verfahren die Semantik der Prädikatenlogik genau einfängt:

- Das Verfahren ist korrekt (es leitet nur echte Tautologien ab) – das war bekannt
- Das Verfahren ist vollständig (es kann jede Tautologie auch irgendwie ableiten) – ein Durchbruch, Gödels Vollständigkeitssatz

Theorie und Praxis

Gödel zeigte damit auch die Semientscheidbarkeit des logischen Schließens:

- Man kann systematisch alle möglichen Ableitungen neuer Tautologien bilden
- Eine Formel ist eine Tautologie genau dann wenn sie irgendwann abgeleitet wird
- Ist eine Formel keine Tautologie, dann werden wir es auf diese Art nie erfahren

Das ist nicht sehr praktisch, wenn man wissen will, ob eine gegebene Formel F eine Tautologie ist . . .

Geht es besser?

- Logischen Schließens ist unentscheidbar: kein Verfahren kann die Frage nach Allgemeingültigkeit sicher beantworten
- Aber: Es gibt effizientere, zielgerichtetere Methoden

Für echte Implementierungen (Jahrzehnte später!) wurden bessere Algorithmen entwickelt

Bessere Algorithmen

Ein besonders erfolgreicher Algorithmus in praktischen Implementierungen ist Resolution

Rückblick: Resolution in Aussagenlogik (Formale Systeme, V.23)

- Methode zum Test auf Unerfüllbarkeit
- Die Formel wird zunächst umgeformt in Klauselform (KNF) = konjunktive Normalform in Mengenschreibweise
- In jedem Ableitungsschritt leitet man aus zwei zueinander passenden Klauseln eine neue ab:

$$\frac{\{L_1, \ldots, L_n, A\} \qquad \{\neg A, L'_1, \ldots, L'_m\}}{\{L_1, \ldots, L_n, L'_1, \ldots, L'_m\}}$$

- Das Verfahren endet, wenn man entweder die leere Klausel erhält (unerfüllbar) oder, andernfalls, wenn keine neuen Klauseln mehr erzeugt werden können (erfüllbar)

Resolution in Prädikatenlogik

Resolution in der Prädikatenlogik funktioniert ganz ähnlich:

(1) Umformung in Klauselform

(2) Durchführung von Resolutionsschritten, bei denen jeweils zwei Klauseln kombiniert werden

(3) Terminierung wenn leere Klausel auftritt oder keine neuen Klauseln mehr entstehen

Wir müssen aber mehr beachten als zuvor:

- Die Umformung in Klauselform muss jetzt Quantoren berücksichtigen
- Die Resolutionsschritte müssen die kompliziertere Form von Atomen (Prädikate mit Termen als Parameter) berücksichtigen
- Es wird im Allgemeinen möglich sein, unendlich viele Klauseln abzuleiten ohne jemals die leere Klausel zu erzeugen – Terminierung nicht garantiert

Syntaktische Umformungen in der Prädikatenlogik

Formeln umformen

Wir haben bereits gelernt (Vorlesung 14):

- In prädikatenlogischen Formeln darf man Teilformeln durch semantisch äquivalente Formeln ersetzten (Ersetzungstheorem)
- Für die aussagenlogischen Junktoren gelten die gleichen Äquivalenzen wie in der Aussagenlogik

$\leadsto$ damit kann man schon viele Umformungen vornehmen

Es fehlen uns aber noch Äquivalenzen zum Umgang mit Quantoren

Äquivalenzen mit Quantoren

Es gelten die folgenen Beziehungen:

$$\neg\exists x.F \equiv \forall x.\neg F$$
$$\neg\forall x.F \equiv \exists x.\neg F$$

Negation von Quantoren

$$\exists x.\exists y.F \equiv \exists y.\exists x.F$$
$$\forall x.\forall y.F \equiv \forall y.\forall x.F$$

Kommutativität

$$\exists x.(F \vee G) \equiv (\exists x.F \vee \exists x.G)$$

Distributitivität $\exists/\vee$

$$\forall x.(F \wedge G) \equiv (\forall x.F \wedge \forall x.G)$$

Distributitivität $\forall/\wedge$

Beweis: Die Beweise ergeben sich direkt aus der Semantik, z.B.:

$\mathcal{I}, \mathcal{Z} \models \neg\exists x.F$ gdw. $\mathcal{I}, \mathcal{Z} \not\models \exists x.F$
gdw. es gibt kein $\delta \in \Delta^{\mathcal{I}}$ mit $\mathcal{I}, \mathcal{Z}[x \mapsto \delta] \models F$
gdw. für alle $\delta \in \Delta^{\mathcal{I}}$ gilt $\mathcal{I}, \mathcal{Z}[x \mapsto \delta] \not\models F$
gdw. für alle $\delta \in \Delta^{\mathcal{I}}$ gilt $\mathcal{I}, \mathcal{Z}[x \mapsto \delta] \models \neg F$
gdw. $\mathcal{I}, \mathcal{Z} \models \forall x.\neg F$ □

Nicht-Äquivalenzen mit Quantoren

Andere ähnliche Beziehungen gelten nicht.

Es gilt nicht $\exists x.\forall y.F \equiv \forall y.\exists x.F$!

Zum Beispiel bedeutet $\forall x.\exists y.\text{hatVater}(x, y)$ „Jeder hat einen Vater“ dagegen $\exists y.\forall x.\text{hatVater}(x, y)$ „Jemand ist Vater aller.“

(Anmerkung: Gödel war übrigens religiös und hat einen philosophischen Gottesbeweis in modaler Logik formalisiert [siehe „Gödel's ontological proof“]; Russell war engagierter Atheist und scharfer Kritiker der Kirchen [siehe z.B. „Why I am not a Christian,“ 1927; deutsche Übersetzung erstmals 1932 in Dresden veröffentlicht].)

Es gilt nicht $\forall x.(F \vee G) \equiv (\forall x.F \vee \forall x.G)$!

Zum Beispiel ist $\forall x.(\text{glücklich}(x) \vee \neg\text{glücklich}(x))$ („Jeder ist entweder glücklich oder nicht.“) eine Tautologie, $(\forall x.\text{glücklich}(x) \vee \forall x.\neg\text{glücklich}(x))$ („Entweder sind alle glücklich oder keiner.“) dagegen nicht.

Negationsnormalform

Eine Formel F ist in Negationsnormalform (NNF) wenn

(a) sie nur Quantoren und die Junktoren $\wedge$, $\vee$ und $\neg$ enthält und

(b) der Junktor $\neg$ nur direkt vor Atomen vorkommt.

Formeln, die negierte oder nicht-negierte Atome sind, nennt man Literale. In NNF darf Negation also nur in Literalen auftauchen.

Beispiele:

- $\forall x.(p(x) \wedge \exists y.(\neg r(x,y) \vee q(y)))$ ist in NNF
- $\neg\forall x.p(x)$ ist nicht in NNF
- $\exists x.\neg\neg p(x)$ ist nicht in NNF
- $\exists x.(p(x) \leftrightarrow p(x))$ ist nicht in NNF

Umwandlung in NNF (1)

Es ist möglich, eine Formel rekursiv in NNF umzuformen.

Dazu ersetzten wir zunächst alle Vorkommen von $\rightarrow$ und $\leftrightarrow$ unter Verwendung der bekannten Äquivalenzen:

$$(F \rightarrow G) \equiv (\neg F \vee G)$$
$$(F \leftrightarrow G) \equiv \big((\neg F \vee G) \wedge (\neg G \vee F)\big)$$

Umwandlung in NNF (2)

Sei F eine Formel, die nur Quantoren und die Junktoren $\wedge$, $\vee$ und $\neg$ enthält. Wir definieren eine Formel $\mathrm{NNF}(F)$ rekursiv wie folgt:

- $\mathrm{NNF}(F) = F$ falls F Atom ist
- $\mathrm{NNF}(F \wedge G) = \mathrm{NNF}(F) \wedge \mathrm{NNF}(G)$
- $\mathrm{NNF}(F \vee G) = \mathrm{NNF}(F) \vee \mathrm{NNF}(G)$
- $\mathrm{NNF}(\exists x.F) = \exists x.\mathrm{NNF}(F)$
- $\mathrm{NNF}(\forall x.F) = \forall x.\mathrm{NNF}(F)$
- $\mathrm{NNF}(\neg F) = \neg F$ falls F Atom ist
- $\mathrm{NNF}(\neg\neg F) = \mathrm{NNF}(F)$
- $\mathrm{NNF}(\neg(F \wedge G)) = \mathrm{NNF}(\neg F) \vee \mathrm{NNF}(\neg G)$
- $\mathrm{NNF}(\neg(F \vee G)) = \mathrm{NNF}(\neg F) \wedge \mathrm{NNF}(\neg G)$
- $\mathrm{NNF}(\neg\exists x.F) = \forall x.\mathrm{NNF}(\neg F)$
- $\mathrm{NNF}(\neg\forall x.F) = \exists x.\mathrm{NNF}(\neg F)$

Quantoren, die nichts binden

Offensichtlich gilt

$$\exists x.F \equiv F \equiv \forall x.F \qquad \text{falls } x \text{ in } F \text{ nicht als freie Variable vorkommt}$$

weil die Zuweisung von x bei der Frage $\mathcal{I}, \mathcal{Z} \overset{?}{\models} F$ keine Rolle spielt.

Dieses Prinzip funktioniert in vielen Fällen.

Kommt x in F nicht als freie Variable vor, dann gilt:

$$\exists x.(F \wedge G) \equiv (F \wedge \exists x.G) \equiv (\exists x.F \wedge \exists x.G)$$
$$\forall x.(F \vee G) \equiv (F \vee \forall x.G) \equiv (\forall x.F \vee \forall x.G)$$
$$\exists x.\forall y.F \equiv \forall y.\exists x.F$$
$$\forall x.\exists y.F \equiv \exists y.\forall x.F$$

Variablenumbenennung

Man kann durch Quantoren gebundene Vorkommen von Variablen einheitlich umbenennen, ohne die Semantik der Formel zu ändern.

Satz: Sei $Qx.F$ mit $Q \in \{\exists, \forall\}$ eine Formel, sei y eine Variable, die nicht in F vorkommt, und sei $F\{x \mapsto y\}$ die Formel, die man erhält, wenn man alle freien Vorkommen von x in F durch y ersetzt. Dann gilt $Qx.F \equiv Qy.F\{x \mapsto y\}$.

Es ist aber wichtig, dass die eingesetzte Variable nicht schon in F vorkommt (auch nicht gebunden):

Beispiel: Sei G die Formel $\forall x.\exists y.\text{hatVater}(x, y)$. Wir können x in z umbenennen und erhalten die semantisch äquivalente Formel $\forall z.\exists y.\text{hatVater}(z, y)$. Benennen wir x dagegen in y um, dann entsteht $\forall y.\exists y.\text{hatVater}(y, y)$ („Jemand ist sein eigener Vater“).

Bereinigte Formeln

Eine Formel F ist bereinigt, wenn sie die folgenden beiden Eigenschaften hat:

(1) Keine Variable in F kommt sowohl frei als auch gebunden vor

(2) Keine Variable in F wird in mehr als einem Quantor gebunden

Man kann jede Formel leicht durch Umbenennung gebundener Variablen bereinigen.

Beispiel: Die Formel

$$\forall y.(p(x,y) \rightarrow \exists x.(r(y,x) \wedge \forall y.q(x,y)))$$

kann wie folgt bereinigt werden:

$$\forall y.(p(x,y) \rightarrow \exists z.(r(y,z) \wedge \forall v.q(z,v)))$$

Zusammenfassung und Ausblick

Logisches Schließen in Prädikatenlogik ist unentscheidbar (gezeigt) aber semi-entscheidbar (noch zu zeigen)

Resolution kann auch in der Prädikatenlogik zum logischen Schließen eingesetzt werden

Mithilfe semantischer Äquivalenzen kann man beliebige Formeln in einheitliche Normalformen überführen.

Was erwartet uns als nächstes?

- Funktionen
- Resolution
- Logik über endlichen Modellen und ihre praktische Anwendung

Bildrechte

Folie 11: Fotografie um 1926, gemeinfrei