TECHNISCHE UNIVERSITÄT DRESDEN

THEORETISCHE INFORMATIK UND LOGIK

21. Vorlesung: Endliche Modelle/Gödels Sätze

Markus Krötzsch
Lehrstuhl Wissensbasierte Systeme

TU Dresden, 5. Juli 2017

Logik und Datenbanken

Relationale Datenbankinstanzen = Endliche Interpretationen

- Tabellen(namen) entsprechen Prädikatssymbolen
- Kleinere syntaktische Unterschiede (benannte vs. geordnete Parameter)

Relationale Datenbankanfragen = Prädikatenlogische Formeln

- Zuweisungen $\mathcal{Z}$ zu freien Variablen als mögliche Anfrageergebnisse
- $\mathcal{I}, \mathcal{Z} \models Q$ bedeutet: $\mathcal{Z}$ ist Ergebnis der Anfrage Q auf Datenbankinstanz $\mathcal{I}$
- Prädikatenlogik kann die selben Anfragefunktionen ausdrücken wie normales SQL

Relationale Algebren

Datenbankanfragen werden oft in relationaler Algebra dargestellt, bei der man Relationen mit Operationen zu einem Anfrageergebnis kombiniert

Beispiel: Die Anfrage

$$Q = \exists z_{\text{Linie}}.(\text{verbindung}(x_{\text{Von}}, x_{\text{Zu}}, z_{\text{Linie}}) \wedge \text{linien}(z_{\text{Linie}}, x_{\text{Typ}}))$$

entspricht einer (natürlichen) Join-Operation ($\wedge$) mit anschließender Projektion ($\exists$):

$$\pi_{\text{Von,Zu,Linie}}(\text{verbindung} \bowtie \text{linien})$$

Anmerkung: SQL hat noch einen leicht anderen Stil. Variablen stehen dort für ganze Tabellenzeilen und man verwendet Notation der Form „linien.Typ“, um auf deren Einträge zuzugreifen („Tuple-Relational Calculus“). Das ändert an der Ausdrucksstärke nichts.

Anfragebeantwortung als Modell Checking

Erkenntnis: Die wesentliche Berechnungsaufgabe bei der Beantwortung von Datenbankabfragen ist das folgende Entscheidungsproblem:

Das Auswertungsproblem (Model Checking) der Prädikatenlogik lautet wie folgt:

Gegeben: Eine Formel Q mit freien Variablen $x_1, \dots, x_n$; eine endliche Interpretation $\mathcal{I}$; Elemente $\delta_1, \dots, \delta_n \in \Delta^{\mathcal{I}}$
Frage: Gilt $\mathcal{I}, \{x_1 \mapsto \delta_1, \dots, x_n \mapsto \delta_n\} \models Q$?

Naive Methode der Anfragebeantwortung:

- Betrachte alle $(\Delta^{\mathcal{I}})^n$ möglichen Ergebnisse
- Entscheide jeweils das Auswertungsproblem

Praktisch relevante Frage:

Wie schwer ist das Auswertungsproblem?

Ein Algorithmus für das Auswertungsproblem

Wir nehmen an, dass die Formel F nur $\neg$, $\wedge$ und $\exists$ enthält (durch Normalisierung möglich)

```
function Eval(F, I, Z) :
01  switch (F) :
02    case p(c_1, ..., c_n) : return ⟨c_1^{I,Z}, ..., c_n^{I,Z}⟩ ∈ p^I
03    case ¬G : return not Eval(G, I, Z)
04    case G_1 ∧ G_2 : return Eval(G_1, I, Z) and Eval(G_2, I, Z)
05    case ∃x.G :
06      for c ∈ Δ^I :
07        if Eval(G{x ↦ c}, I, Z) then return true
08      return false
```

Anmerkung: Wenn Konstanten c in der Anfrage vorkommen, dann nimmt man in der Regel an, dass $c^{\mathcal{I}} = c$ ist.

Zeitkomplexität

Sei m die Größe von F und $n = |\mathcal{I}|$ (Gesamtgröße der Datenbank)

- Wie viele rekursive Aufrufe von Eval gibt es?
  $\rightsquigarrow$ einen pro Teilformel: $\leq m$
- Maximale Rekursionstiefe?
  $\rightsquigarrow$ beschränkt durch Gesamtzahl der Aufrufe: $\leq m$
- Maximale Zahl der Iterationen in **for**-Schleife?
  $\rightsquigarrow |\Delta^{\mathcal{I}}| \leq n$ pro rekursivem Aufruf
  $\rightsquigarrow$ insgesamt $\leq n^m$ Iterationen
- $\langle c_1^{\mathcal{I},\mathcal{Z}}, \ldots, c_n^{\mathcal{I},\mathcal{Z}} \rangle \in p^{\mathcal{I}}$ ist entscheidbar in linearer Zeit bzgl. n

Gesamtlaufzeit in $m \cdot n^m \cdot n = m \cdot n^{m+1}$:

- Komplexität des Algorithmus: in ExpTime
- Komplexität bzgl. Größe der Datenbank (m konstant): in P

Speicherkomplexität

Wir erhalten eine bessere Komplexitätsabschätzung, wenn wir den Speicherbedarf betrachten

Sei m die Größe von F und $n = |\mathcal{I}|$ (Gesamtgröße der Datenbank)

- Speichere pro (rekursivem) Aufruf einen Pointer auf eine Teilformel von F: $\log m$
- Speichere für jede Variable in F (maximal m) die aktuelle Zuweisung (als Pointer): $m \cdot \log n$
- $\langle c_1^{\mathcal{I},\mathcal{Z}}, \ldots, c_n^{\mathcal{I},\mathcal{Z}} \rangle \in p^{\mathcal{I}}$ ist entscheidbar in logarithmischem Speicher bzgl. n

Speicher in $m \log m + m \log n + \log n = m \log m + (m+1) \log n$

- Komplexität des Algorithmus: in PSpace
- Komplexität bzgl. Größe der Datenbank (m konstant): in L

Zur Erinnerung: PSpace $\subseteq$ ExpTime und L $\subseteq$ P, d.h. die obigen Schranken sind besser

Komplexität des Auswertungsproblems

Satz: Das Auswertungsproblem der Prädikatenlogik ist PSpace-vollständig.

Beweis: Durch Reduktion vom Auswertungsproblem quantifizierter Boolscher Formeln (**TrueQBF**).

Sei $Q_1 p_1 . Q_2 p_2 . \cdots Q_n p_n . F[p_1, \ldots, p_n]$ eine QBF (mit $Q_i \in \{\forall, \exists\}$)

- Datenbankinstanz $\mathcal{I}$ mit $\Delta^{\mathcal{I}} = \{0, 1\}$
- Eine Tabelle mit einer Spalte: $\text{true}(1)$
- Aus der gegebenen QBF erstellen wir die folgende prädikatenlogische Formel ohne freie Variablen:

$$Q_1 x_1 . Q_2 x_2 . \cdots Q_n x_n . F[p_1/\text{true}(x_1), \ldots, p_n/\text{true}(x_n)]$$

  wobei $F[p_1/\text{true}(x_1), \ldots, p_n/\text{true}(x_n)]$ die Formel ist, die aus F entsteht, wenn man jedes aussagenlogische Atom p_i durch das prädikatenlogische Atom $\text{true}(x_n)$ ersetzt.

Die Korrektheit dieser Reduktion ist leicht zu zeigen. □

Wie schwer sind Datenbankabfragen?

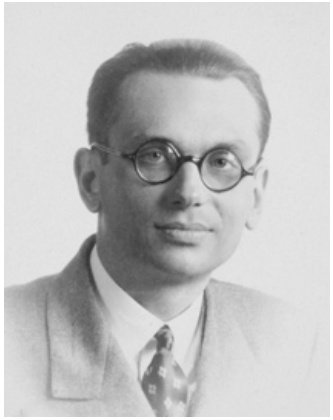
Korollar: Die Beantwortung von SQL-Anfragen ist PSpace-hart, sogar wenn die Datenbank nur eine einzige Tabelle mit einer einzigen Zeile enthält.

Die Komplexität steckt vor allem in der Struktur der Anfrage.

Ist die Anfrage fest vorgegeben oder in ihrer Größe beschränkt, dann wird die Komplexität von der Datenbankgröße dominiert: bezüglich dieser Größe ist das Problem aber in L.

Man kann sogar noch niedrigere Komplexitätsschranken bzgl. der Datenbankgröße angeben.

$\rightsquigarrow$ SQL-Anfragebeantwortung ist praktisch implementierbar, aber nur solange die Anfragen nicht zu komplex werden.



Kurt Gödel

Der 1. Gödelsche Unvollständigkeitssatz

Was Gödel in 1931 zeigte war grob gesagt folgendes:

Satz: Jedes formale System, in dem eine gewisse Menge elementarer Arithmetik dargestellt werden kann, ist unvollständig in Bezug auf die Beweisbarkeit von Sätzen der elementaren Arithmetik: Es gibt solche Sätze, die weder bewiesen noch widerlegt werden können.

Um das zu verstehen müssen wir einiges klären:

- Was ist ein formales System?
- Was ist „eine gewisse Menge elementarer Arithmetik“?
- Was genau bedeutet unvollständig hier?

Formale Systeme

Ein formales System ist ganz allgemein ein Beweissystem, bestehend aus:

- Einer Sprache, in der Aussagen formuliert werden können
- Einer Menge von Axiomen, d.h. als wahr vorgegebener Aussagen
- Einem effektiven Verfahren mit dem man aus gegebenen Aussagen neue Schlüsse ableiten kann

Beweisbare Sätze heißen Theoreme des formalen Systems

Anmerkung: Auch die Axiome sind Theoreme, wenn auch mit sehr kurzen Beweisen

Beispiel: Prädikatenlogik liefert formales Systeme:

- Sprache: Die Sprache der prädikatenlogischen Sätze
- Axiome: Eine gegebene Theorie, z.B. die Theorie der kommutativen Monoide
- Ableitungsverfahren: Resolutionskalkül

Formale Systeme: Wichtige Eigenschaften

Formale Systeme können viele Formen haben – zum Beispiel beinhalten sie auch jedes System, in dem mathematische Beweise formal geführt werden können (z.B. ZFC: Mengenlehre nach Zermelo-Fraenkel mit Auswahlaxiom).

Die Details sind relativ egal, aber wir wollen doch ein paar Anforderungen stellen:

Grundeigenschaft Formaler Systeme: Die Menge der Theoreme eines formalen Systems ist rekursiv aufzählbar (semi-entscheidbar).

Negation: Für jeden Satz S gibt es in der Sprache eines formalen Systems auch einen negierten Satz $\neg S$, so dass gilt:

- S ist genau dann ein Theorem, wenn $\neg\neg S$ ein Theorem ist.
- Wenn S und $\neg S$ Theoreme sind, dann sind alle Sätze Theoreme.

Eigenschaften formaler Systeme

Syntaktische Eigenschaften:

- Konsistenz: Ein formales System ist konsistent, wenn es keinen Satz S gibt, so dass S und $\neg S$ beweisbar sind.
- Vollständigkeit: Ein formales System ist (negations-)vollständig, wenn für jeden Satz S entweder S oder $\neg S$ beweisbar sind.

Wenn man z.B. in Prädikatenlogik arbeitet, dann kann man Mengen von Sätzen eine Semantik (Modelltheorie) geben und weitere Eigenschaften fordern:

Semantische Eigenschaften:

- Korrektheit: Ein formales System ist korrekt, wenn jeder beweisbare Satz auch semantisch wahr (tautologisch) ist.
- Vollständigkeit: Ein formales System ist (semantisch) vollständig, wenn jeder semantisch wahre Satz beweisbar ist.

Verwechslungsgefahr

Achtung! Es gibt zwei Arten von Vollständigkeit:

- Negationsvollständigkeit: die syntaktische Eigenschaft, dass jeder Satz bewiesen oder widerlegt werden kann
- Semantische Vollständigkeit: die semantische Eigenschaft, dass jede Tautologie bewiesen werden kann

Gödels Vollständigkeitssatz bezieht sich auf die zweite Art von Vollständigkeit, Gödels Unvollständigkeitssätze auf die erste!

Beispiele

Beispiel: Angenommen ein Satz F kann in einem formalen System $\mathcal{S}$ bewiesen und wir wissen, dass $\mathcal{S}$ konsistent ist. Folgt daraus, dass F semantisch wahr ist?

Beispiel: Aus der Logik bekannte Zusammenhänge gelten auch hier:

- Ein Satz F ist genau dann in $\mathcal{S}$ beweisbar wenn $\mathcal{S}$ bei Hinzunahme des Axioms $\neg F$ inkonsistent wird.
- Weder F not $\neg F$ sind in $\mathcal{S}$ beweisbar gdw. $\mathcal{S}$ sowohl bei Hinzunahme von F als auch bei Hinzunahme von $\neg F$ konsistent ist

Eine gewisse Menge Arithmetik

Satz: Jedes formale System, in dem eine gewisse Menge elementarer Arithmetik dargestellt werden kann, ist unvollständig in Bezug auf die Beweisbarkeit von Sätzen der elementaren Arithmetik: Es gibt solche Sätze, die weder bewiesen noch widerlegt werden können.

Was ist mit „eine gewisse Menge elementarer Arithmetik“ gemeint?

- Das System sollte Sätze über Beziehungen von konkreten natürlichen Zahlen ausdrücken können
- Dabei sollten die elementaren Operationen $+$, $-$ und $\times$ sowie die Relation $=$ unterstützt werden
- Das System sollte bezüglich der gängigen Semantik dieser arithmetischen Ausdrucksmittel korrekt sein

Oft kommt man mit noch weniger aus, aber diese Eigenschaften reichen in der Regel

Arithmetik in Prädikatenlogik

Man kann die benötigte Menge von Arithmetik durch eine prädikatenlogische Theorie axiomatisieren:

- Konstante $\mathbb{0}$
- Funktionssymbole s (unär, „Nachfolger“), $+$ und $\times$ (binär, infix)
- Prädikatssymbol $\approx$ (binär, infix)

Darstellung natürlicher Zahlen als Nachfolger von 0:

$$\mathbb{0} \hat{=} 0,\ s(\mathbb{0}) \hat{=} 1,\ s(s(\mathbb{0})) \hat{=} 2,\ \ldots$$

Sätze zur Axiomaitiserung der Grundrechenarten:

- $\forall x.\neg(s(x) \approx \mathbb{0})$
- $\forall x.(x + \mathbb{0} \approx x)$
- $\forall x, y.(x + s(y) \approx s(x + y))$
- $\forall x.(x \times \mathbb{0} \approx \mathbb{0})$
- $\forall x, y.(x \times s(y) \approx (x \times y) + x)$
- . . . (mit obigem kann man schon einiges ausrechnen, aber es gibt noch mehr zu sagen, z.B. eine komplette Gleichheitstheorie)

Gödels Beweis des 1. Satzes

Idee

Wie kann man zeigen, dass irgendein Satz weder beweisbar noch widerlegbar ist?

Ein einfaches Szenario:

- Annahme: Sie glauben nur Dinge, die wirklich wahr sind, und was Sie glauben ist konsistent.
- Behauptung: Es gibt wahre Sätze, die sie nicht glauben.
- Beweis: „Ich mache jetzt eine wahre Behauptung, die Sie mir nicht glauben“ ist eine Behauptung, die sie nicht glauben können:
  - Wenn Sie sie glauben, dann muss sie wahr sein, d.h. Sie glauben sie nicht – das wäre inkonsistent
  - Also glauben Sie sie nicht. Dann ist die Behauptung wahr. □

Behauptungen dieser Form nennt man Gödelsätze

Gödelsätze formal machen

Natürliche Sprache eignet sich nicht für stichhaltige Argumente, da sie keine klar definierte mathematische Interpretation hat (z.B. kann ich sagen „Dieser Satz ist eine Lüge“)

Gödels Beweis (vereinfacht) für formale Systeme:

- Annahme: Das gegebene System ist korrekt und konsistent.
- Behauptung: Es gibt wahre Sätze, die nicht beweisbar sind.
- Beweis: Gödel definiert eine mathematische Formel F, welche ausdrückt:

  „F ist wahr genau dann wenn F nicht beweisbar ist.“

  – Wenn F beweisbar wäre, dann ist es wahr und also nicht beweisbar – Widerspruch
  – Also ist F nicht beweisbar, und damit wahr □

Gödels Zahlen

„F ist wahr genau dann wenn F nicht beweisbar ist.“

Die Herausforderung ist, solche Gödensätze zu definieren: Offenbar kann man F nicht als Teilausdruck in F verwenden!

Gödel definiert daher nummerische Bezeichner für Formeln – sogenannte Gödelzahlen – und kodiert seine Sätze anders:

„F ist wahr genau dann wenn m in der Menge der Gödelzahlen nicht beweisbarer Sätze vorkommt.“

wobei m die Gödelzahl für F ist.

Man muss dafür eine Menge technischer Ergebnisse zeigen, z.B.

- Ein solcher Satz F, welcher seine eigene Gödelzahl verwendet, existiert überhaupt
- Die Menge der Gödelzahlen nicht beweisbarer Sätze ist arithmetisch darstellbar

Quotes und Quines

Eine verwandte Form von Selbstbezüglichkeit ist auch in der Informatik bekannt:

Ein Quine ist ein Programm, das bei einer leeren Eingabe seinen eigenen Quellcode ausgibt.

Auch hier denkt man vielleicht zuerst, dass dies nicht möglich wäre, weil da Programm dazu seinen eigenen Code enthalten müsste – eine unendliche Rekursion . . .

Es ist aber gar nicht so schwer:

Beispiel: Gib den folgenden Satz zweimal aus, beim zweiten mal in Anführungszeichen: "Gib den folgenden Satz zweimal aus, beim zweiten mal in Anführungszeichen: "

Gödels 1. Satz berechnungstheoretisch zeigen

Unentscheidbarkeit und Unvollständigkeit

Wir wissen:

Die Theoreme formaler Systeme sind rekursiv aufzählbar, d.h. semi-entscheidbar

- Speziell gilt das auch für die Theoreme, die sich auf arithmetische Aussagen beziehen

Es gibt Mengen (Sprachen), die nicht semi-entscheidbar sind

- Zum Beispiel das Nicht-Halteproblem von Turingmaschinen

⇝ Falls man Instanzen des Wortproblems einer nicht-semientscheidbaren Sprache auf die Wahrheit arithmetischer Sätze reduzieren könnte, dann würde daraus schon Unvollständigkeit folgen

Zahlen vs. Sprachen

Berechnungstheorie handelt von Sprachen, d.h. Mengen von Wörtern

Es ist leicht möglich, jedem Wort eine natürliche Zahl zuzuordnen – man spricht von einer Gödelzahl für das Wort

(dies ist möglich, da die Menge aller Wörter abzählbar ist, siehe Formale Systeme WS 2016/2017; natürlich gibt es viele mögliche Zuordnungen).

Wir folgern:

- Es gibt eine Eins-zu-eins-Beziehung zwischen formalen Sprachen (über einem gegebenen Alphabet) und Mengen natürlicher Zahlen
- Wir können also von der Entscheidbarkeit/Unentscheidbarkeit einer Menge natürlicher Zahlen sprechen
- Es gibt unentscheidbare Mengen natürlicher Zahlen, z.B. die Menge der Zahlen, welche ein Wort enummieren, das eine Turingmaschine kodiert, welche auf der leeren Eingabe hält

Eine erste Beweisidee

Mögliche Strategie zum Beweis von Gödels 1. Satz:

- Definiere eine Gödel-Nummerierung für alle Wörter die eine Turingmaschine kodieren
- Finde arithmetische Ausdrücke auf diesen Zahlen, mit denen man konkrete syntaktische Eigenschaften der kodierten TM testen kann
- Finde arithmetische Formeln, die ausdrücken, dass die kodierte TM auf dem leeren Wort hält

⇝ Programmiere eine universelle TM in Arithmetik

(Machbar, aber aufwändig!)

Ein schweres arithmetisches Problem

Die Aufgabe wird viel einfacher, wenn man mit ein unentscheidbares Problem verwendet, welches sich schon auf arithmetische Aussagen bezieht.

Eine Diophantische Gleichung ist ein arithmetischer Ausdruck der Form $D[x_1, \ldots, x_n] = 0$, wobei $D[x_1, \ldots, x_n]$ ein Funktionsterm über den Symbolen $x_1, \ldots, x_n, 0, s, +, -, \times$ ist. Eine Lösung der Gleichung ist eine Liste von natürlichen Zahlen $z_1, \ldots, z_n$, für welche die Gleichung stimmt.

Beispiele für diophantische Gleichungen:

- $y = 3x + 5$, d.h. $3 \times x - y + 5 = 0$ (unendlich viele Lösungen)
- $x^2 + y^2 = -1$, d.h. $x \times x + y \times y + 1 = 0$ (keine Lösung)
- $x^2 + y^2 = z^2$ (unendlich viele Lösungen)
- $x^4 + y^4 = z^4$ (genau eine Lösung, Fermatsche Vermutung)
- $x^2 = 2y^4 - 1$ (genau zwei Lösungen: 1 & 1 und 13 & 239)

Hilberts Zehntes und MRDP

Hilberts 10. Problem: Man finde ein Verfahren um zu ermitteln, ob eine gegebene diophantische Gleichung lösbar ist.

Wir wissen heute, dass dies unentscheidbar ist:

- Jede diophantische Gleichung $D[x_1, \ldots, x_n, y] = 0$ definiert eine Menge natürlicher Zahlen $\{k \mid D[x_1, \ldots, x_n, k] = 0 \text{ ist lösbar}\}$
- Eine so definierte Menge heißt diophantische Menge
- Alle diophantischen Mengen sind rekursiv aufzählbar: teste systematisch alle möglichen Belegungen von $D[x_1, \ldots, x_n, y] = 0$ und gib bei jeder gefundenen Lösung den Wert von y aus

Satz (Matiyasevich/Robinson/Davis/Putnam): Jede rekursiv aufzählbare Menge natürlicher Zahlen ist diophantisch.

Intuitiv: diophantische Gleichungen sind Turing-mächtig!

Konsequenzen

Wir folgern:

- Hilberts 10. Problem ist wirklich unentscheidbar:
  - Wäre es entscheidbar, dann könnte man jede diophantische Menge entscheiden (Kontrollfrage: Wie?)
  - Wir wissen aber bereits, dass es rekursiv aufzählbare Mengen natürlicher Zahlen gibt, die nicht entscheidbar sind
- Nicht-Lösbarkeit diophantischer Gleichungen ist nicht semi-entscheidbar:
  - Lösbarkeit ist bereits semi-entscheidbar
  - Wenn Nicht-Lösbarkeit auch semi-entscheidbar wäre, dann wäre beides entscheidbar

Beweis von Gödels 1. Unvollständigkeitssatz

Sei $\mathcal{S}$ ein formales System, welches das folgende erfüllt:

(1) $\mathcal{S}$ ist konsistent

(2) Jede wahre Aussage der Form $D[z_1, \ldots, z_n] = 0$ für eine beliebige diophantische Gleichungen und konkrete Zahlenwerte $z_1, \ldots, z_n$ ist beweisbar (einfach durch Ausrechnen des Wertes)

(3) Sätze der Form $\exists x_1, \ldots, x_n . D[x_1, \ldots, x_n] = 0$ sind darstellbar und alle beweisbaren Sätze dieser Form sind korrekt

Wegen (1) und (2) gilt: $\mathcal{S}$ beweist nur wahre Sätze der Form $\neg\exists x_1, \ldots, x_n . D[x_1, \ldots, x_n] = 0$

Aber: $\mathcal{S}$ kann nicht alle wahren Sätze dieser Form beweisen:

- Die Menge aller beweisbaren Sätze dieser Form ist semi-entscheidbar
- Die Menge aller wahren Sätze dieser Form ist nicht semi-entscheidbar

Mit (3) gilt: Es gibt Sätze F der Form $\exists x_1, \ldots, x_n . D[x_1, \ldots, x_n] = 0$, so dass weder F noch $\neg F$ bewiesen wird. □

Zusammenfassung und Ausblick

Das Auswertungsproblem der Prädikatenlogik ist PSpace-vollständig

Formale Systeme, die elementare Rechnungen auf natürlichen Zahlen ausführen können sind entweder inkonsistent oder (negations-)unvollständig

Diese Unvollständigkeit hat damit zu tun, dass nicht alle Mengen natürlicher Zahlen Turing-erkennbar sind

Was erwartet uns als nächstes?

- 3. Repetitorium
- Probeklausur
- Gödels zweiter Unvollständigkeitssatz, Zusammenfassung und Ausblick

Bildrechte

Folie 10: Fotografie um 1926, gemeinfrei