2. LOGISCHE KALKÜLE FÜR DIE MATHEMATIK

2.1. ÜBERBLICK / MOTIVATION

Wie funktioniert ein log. Kalkül:

(a) Festlegen einer Sprache (entspricht Menge von Zeichenketten)

(b) Beschreibung der Semantik: Was bedeutet ein Satz der Sprache (als Aussage über math. Strukturen) Das führt automatisch zur Definition von "Satz φ ist in einer bestimmten Struktur wahr", und zur semantischen Folgerung: φ folgt aus Satzmenge Σ, wenn jede Struktur die Σ erfüllt auch φ erfüllt. Man schreibt $\Sigma \models \varphi$

(c) Völlig getrennt davon: Definition des logischen Kalküls: "Mechanische" Ableitungsregeln, d.h. "algorithmische" Manipulationen von Zeichenketten, so dass man aus Grund-Zeichenketten ("Axiome") Σ weitere Zeichenketten ableiten kann. Das führt zum Begriff der (formalen) Ableitung bzw. Beweis, geschrieben $\Sigma \vdash \varphi$.

(d) Im Idealfall bekommt man: $\Sigma \models \varphi \iff \Sigma \vdash \varphi$ (siehe Vollständigkeitssatz). Damit hat man den "schwierigen" bzw. möglicherweise unlösbaren Begriff der sem. Folgerung auf den einfacheren der formalen Ableitbarkeit reduziert. Es gibt aber auch scharfe Grenzen der Formalisierbarkeit (siehe

Gödel'sche Unvollständigkeitssatz)

Wozu?

Es ist zwar im normalen mathematischen Alltag nicht nötig, zu präzisieren was genau ein mathe Beweis ist. Nötig bzw notwendig ist es aber jedenfalls in folgenden Situationen:

- Um zu zeigen, dass bestimmte Aussagen unentscheidbar sind (weder beweisbar noch widerlegbar),
  muss man natürlich "Beweis" genau definieren
- Z.b: "Ohne AC kann man Existenz von nichtmessbaren Mg nicht beweisen."

- Um bestimmte, besonders abstrakte, Beweise führen zu können (z.B. AC → Zorn)
  ist es nötig, sich klarzumachen
  welche Methoden genau man für den Beweis verwenden darf.
- Nötig zum automatischen Finden oder zumindest Überprüfen von Beweisen ohne ohne

Die Notwendigkeit der formalen Sprache:
Es gibt in gewissen Hinsicht keine universelle logische Sprache. Die formale Sprache (Objektsprache) ist immer schwächer als die Metasprache, in der wir über den Kalkül/die Objektsprache sprechen.

Am besten kann man das mit dem Richard-Paradoxon demonstrieren:
- Es gibt nur $< 100^{1000}$ Sätze mit weniger als 1000 Buchstaben.
- Es sind also nur endl viele natürl. Zahlen mit solchen Sätzen definierbar.
- Sei n die kleinste natürliche Zahl die nicht mit einem deutschen Satz mit weniger als tausend Buchstaben definierbar ist.
- Weil n offenbar durch den vorigen Satz def. wird, ist das ein Widerspruch.

Die Ursache des Widerspruchs ist das nicht, unterschieden sind zwischen der Objektsprache (einer formalen Sprache $\mathcal{L}$) und der Metasprache ("Deutsch", in der formuliert werden kann ob eine Zahl n in $\mathcal{L}$ mit weniger Zeichen definiert werden kann. Man muß also präzisieren:
Definierbar mit welchen Methoden / Sprachen?
Einen universellen Begriff der Definierbarkeit (und eine universelle formale Sprache) gibt es nicht.

Beispiele für Sprachen:

- Aussagenlogik: Bsp: $(A \to B) \to (\neg B \to \neg A)$
  Sehr schwach, aber sogar dieser Teil der math. Logik war vor 100 Jahren recht kontrovers (tertium non datur: $A \vee \neg A$, wurde von Intuitionisten abgelehnt).

- Prädikatenlogik / first order logic:
  Die wichtigste Sprache für die math. Logik:
  $(\forall \varepsilon > 0)(\exists \delta > 0) \quad |x' - x| < \delta \to |f(x) - f(x')| < \varepsilon$
  oder $\forall a \exists b \; a \cdot b = b \cdot a = e$
  kann nicht als first order Logik formuliert werden. Quantoren über nur über "Dinge", nicht über Teilmengen (= Eigenschaften), z.B.
  $(\forall A)(\exists x\, A(x)) \to (\exists x\, A(x) \wedge \forall x (x \to \neg A(x)))$
  ↑ Eigenschaft
  ist sog. <u>second order</u> Logik.
  Die ist zwar ausdrucksstärker, aber so stark, dass sie kein log. Kalkül haben kann, daher nicht den prädikatenlogischen Anforderungen der log. Sprache entspricht.

2.2 AUSSAGENLOGIK

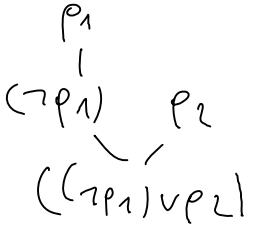
DIE SPRACHE

Wir verwenden folgende Symbole:
- Aussagenlog. Variablen $p_1, p_2, \ldots$
- Junktoren: $\wedge$ (und, Konjunktion)
  $\vee$ (oder, Disjunktion)
  $\rightarrow$ (folgt, Implikation)
  $\neg$ (nicht, Negation)
- Klammern: (und)

Zeichenketten sind endl. Folgen von Symbolen: 2B $\rightarrow$ (oder $(p_1 \vee p_2)$

Bestimmte Zeichenketten sind Formeln:
(a) Alle Variablen sind Folgen
(b) wenn A, B Formeln, dann auch
$(\neg A)$, $(A \rightarrow B)$, $(A \wedge B)$, $(A \vee B)$

(Wenn man pedantisch sein will:
Die Formeln sind die kleinste TM oder Zeichenketten, die (a) enthalten und unter (b) abgeschlossen sind.
Es ist wichtig zu sehen dass Formeln eindeutig lesbar sind: $((\neg p_1) \vee p_2)$
kann nur auf genau eine Weise gebildet worden sein:



SEMANTIK

Eine Belegung ist eine Funktion
b: Variablen $\to \{0,1\}$. (0 ... falsch, 1 ... wahr)
Jede Belegung
lässt sich auf alle Formeln fortsetzen,
mit Induktion nach (eindeutigem) Formel-
Aufbau: Wenn $b(A)$ schon def,
dann sehe $b((\neg A)) = 0$ gdw $b(A) = 1$.
In Tabellenform:

A	$(\neg A)$
0	1
1	0

Genauso wird $b(C)$ für die anderen
Junktoren entsprechend folgender
Tabelle def:

A	B	$\neg A$	$A \wedge B$	$A \vee B$	$A \to B$
0	0	1	0	0	1
0	1	1	0	1	1
1	0	0	0	1	0
1	1	0	1	1	1

~

Bsp: Angenommen, $b(p_1) = 0$ und
$b(p_2) = 1$. Dann ist $b((p_1 \vee p_2) \wedge (\neg p_1)) = 1$

~

Die Wahrheit von A kann sich also in
Abhängigkeit von der Wahrheit der in A
vorkommenden Variablen in Form einer
Wahrheitstabelle ergeben:
Bsp: $A = (p_1 \wedge (\neg p_2))$
(lies: wenn $b(p_1) = 0$
und $b(p_2) = 0$, dann $b(A) = 0$
etc)

p_1	p_2	A
0	0	0
0	1	0
1	0	1
1	1	0

A heißt Tautologie, wenn $b(A) = 1$ für alle Belegungen b.
A heißt erfüllbar, wenn $b(A) = 1$ für eine Belegung b.
Sonst heißt A unerfüllbar.

Entscheidbarkeit und Kalkül

Die Wahrheitstabelle liefert offenbar einen Algorithmus, um zu entscheiden, ob eine Formel A Tautologie, nicht Tautologie aber erfüllbar, oder unerfüllbar ist.

Die Aussagenlogik ist also so einfach (schwach), dass wir oftmals entscheiden können ob ein Satz wahr ist, wir brauchen dafür in diesem Fall keinen Kalkül (d.h. einen Begriff der formalen Ableitung), bzw wir können einen trivialen Kalkül angeben: $\vdash A$ ("A ist ableitbar") gdw A Tautologie.
(Bem: es gibt aber sehr wohl "wichtige" Kalküle für die Aussagenlogik, z.B. Resolutionskalküle oder Hilbert-artige Kalküle wie Frege-Łukasiewicz)

Bem: Um zu testen, ob A Tautologie ist (oder erfüllbar) muss man i.A. 2^n viele Fälle ausprobieren!

wobei n die Anzahl der Variablen ist. (Die Wahrheitstabelle hat ja 2^n viele Zeilen.) Der Entscheidungs-Algorithmus ist also enorm "ineffizient".

Zu entscheiden, ob eine Formel mit 90 Var eine Tautologie ist, braucht bei 2^{30} Tests pro sekunde länger als das Alter des Universums.

Frage: Gibt es besseren Algorithmus?

P/NP-Problem (wichtiges ungelöstes Problem der th. Informatik): Gibt es einen Algorithm. mit polynomieller Laufzeit?

Der "Aussagenlog. Teil" der Mathematik ist aber auch in der Praxis immer trivial, obwohl das Entscheidungs-verfahren so ineffizient ist. In anderen praktischen Fällen ist diese Ineffizienz aber sehr problematisch (automatisches Beweisen, Boolesche Suche in Datenbanken, automatische Verifikation von Schaltkreisen etc).

~

Zwei Formeln A, B sind äquivalent, wenn $b(A) = b(B)$ für jede Belegung b.

Ü: A, B sind äqu.v., gdw $A \leftrightarrow B$ eine Tautologie ist

($A \leftrightarrow B$ ist Abkürzung für $(A \to B) \wedge (B \to A)$)

Disj. Normalform:
A ist in disj. NF, wenn A folgende Form hat: $A = B_1 \vee B_2 \vee \dots \vee B_k$
wobei jedes B_k von der Form $C_i^1 \wedge \dots \wedge C_i^{n_i}$
ist mit C_i^j entweder Variable oder Negation von Variable.
Bsp: $(p_1 \wedge \neg p_2) \vee (p_3 \wedge p_2)$

Es gilt: Zu jeder Formel A gibt es eine äquiv. Formel A' in DNF.

Bew: Ü (Wahrheitstabelle)

~

Vollständigkeit:
Sei A eine Formel mit n Variablen.
Eine Belegung b (eingeschränkt auf die Var. in A) ist also eine Funktion von n nach $\{0,1\}$, in anderen Schreibweise ein Element von 2^n.
Die Wahrheitstabelle von A ist also eine Funktion $G_A : 2^n \to \{0,1\}$, $b \mapsto b(A)$

Es gilt: $\{\wedge, \vee, \to, \neg\}$ sind vollständig, d.h. zu jeder Funktion $G : 2^n \to \{0,1\}$ gibt es ein A mit $G = G_A$.

Beweis: Ü (Wahrheitstabelle)

Ü: Welche TM von $\{\wedge, \vee, \to, \neg\}$ sind vollständig?

Unendliche Formelmengen, semantische Folgerung:

Sei Σ eine Menge von Formeln.
$\Sigma \models A$ heißt: Für alle Belegungen b gilt: wenn $b(B) = 1$ für alle $B \in \Sigma$, dann auch $b(A) = 1$.

Ü Für $\Sigma = \{B_1, \dots, B_n\}$ endlich ist $\Sigma \models A$ äquiv. zu
"$B_1 \wedge \dots \wedge B_n \to A$ ist Tautologie"

Es gilt der <u>Kompaktheitssatz</u> (ohne Beweis):
$\Sigma \models A$ gdw $\Sigma' \models A$ für ein $\Sigma' \subseteq \Sigma$ endlich.

Σ heißt erfüllbar, wenn es ein b gibt so dass $b(B) = 1$ für alle $B \in \Sigma$.
Es gilt: $\Sigma \models A$ gdw $\Sigma \cup \{\neg A\}$ unerfüllbar.
Daher kann man Kompaktheitssatz auch äquivalent schreiben als:
Σ ist erfüllbar gdw jede endl TM von Σ erfüllbar ist.
(Bew der Äquiv.: Ü)
Anwendung: Ü: Eine unendliche Karte ist 3-färbbar gdw jede endliche Teilkarte 3-färbbar ist.