Formale Logik

Wintersemester 2023/24

Dr Dirk Frettlöh
Technische Fakultät
Universität Bielefeld

January 30, 2024

Contents

An den Leser: Dieses Skript wurde ab 2018/19 für die Vorlesung "Formal Logic" an der Uni Bielefeld erstellt. Ab dem Wintersemester 2023/24 findet die Vorlesung auf deutsch statt. Das Skript wird nach und nach vom Englischen ins Deutsche übersetzt. Es finden sich sicher noch etliche (hoffentlich kleinere) Fehler. Wenn Sie einen finden, gerne per Email an mich melden.

1 Aussagenlogik

1.1 Grundlagen

11. Okt.

In der Logik geht es um wahre oder falsche Aussagen. Betrachten wir ein paar Beispiele:

1. Das Universitatshauptgebäude ist schön.
2. $2+2=5$.
3. X ist ein Chinese, also ist X ein Mensch.
4. X ist ein Mensch, also ist X ein Chinese.
5. Bitte nicht rauchen.
6. Hello Kitty.

Nummer 5 and 6 sind keine Aussagen, denn sie sind weder "wahr" noch "falsch". Solche Dinge betrachten wir hier gar nicht.

Aussagen 1-4 *sind* Aussagen, denn sie sind entweder wahr oder falsch. (OK, bei 1 kann man streiten, aber...) Einige der Aussagen 1-4 sind zusammengesetzt, andere nicht. So sind Aussagen 1 und 2 nicht zusammengesetzt: sie lassen sich nicht in kleinere Aussagen zerlegen: jeder kleinere Teil (wie $2+2$, oder "Das Universitatshauptgebäude") sind weder wahr noch falsch, also keine Aussagen. Die Aussagen 3 und 4 sind zusammengesetzt. Beide sind von der Form "A, also B", also sowas wie "aus A folgt B" oder "$A \Rightarrow B$".

Im Rest dieses Abschnitts wollen wir das präzisieren, und zwar, indem wir es abstrahieren (also vom sprachlichen Inhalt trennen) und formalisieren.

Syntax

Eine Aussage, die sich nicht in kleinere Teile zerlegen lässt, heißt **atomare Formel**. Komplexere Aussagen werden aus atomaren Formeln zusammengesetzt, und zwar so:

Definition 1.1. (und Notation) Atomare Formeln werden abgekürzt mit $A, B, C, \ldots$, oder mit $A_1, A_2, \ldots$.

Eine (aussagenlogische) **Formel** ist induktiv definiert durch:

- Jede atomare Formel ist eine Formel.
- Sind F und G Formeln, dann sind $F \vee G$, $F \wedge G$ und $\neg F$ auch Formeln.

Ist H eine Formel, so heißt jedes F und G von oben, aus dem H gebaut wird, **Teilformel** von H.

Alles, was wir so bauen können (und sonst nichts), ist eine Formel. (Genauer: eine aussagenlogische Formel, aber wir lassen das Adjektiv weg, solange es nicht zu Verwechslungen führt; z.B. in diesem ganzen Kapitel. Später lernen wir auch Formeln in anderen Logiken kennn.)

Beispiel 1.2. $F = \neg((A \wedge B) \vee C)$ ist eine Formel. Seine *Teilformeln* sind A, B, C, $A \wedge B$, $(A \wedge B) \vee C$, und $\neg((A \wedge B) \vee C)$. Dagegen sind $(A\wedge$, oder $\vee C$, oder (keine Formeln, also auch keine Teilformeln von F.

Notation

- $A, B, C, \ldots$, or $A_1, A_2, \ldots$ bezeichnen atomare Formeln.
- $F, G, H, \ldots$, or $F_1, F_2, \ldots$ bezeichnen Formeln.
- $F \Rightarrow G$ ist eine Kurzschreibweise für $(\neg F) \vee G$. $F \Leftrightarrow G$ ist eine Kurzschreibweise für $(F \wedge G) \vee (\neg F \wedge \neg G)$.

Später interpretieren wir $\wedge$ als "und" usw. Aber an dieser Stelle ist eine Formel rein abstrakt: die Formeln sind einfach nur eine Aneinanderreihung von Zeichen nach bestimmten Regeln. Das ist die **Syntax**: Symbole und Regeln. Erst im Folgenden geben wir ihnen eine Bedeutung. Das ist die *Semantik*:

Semantik

Definition 1.3. Die Elemente der Menge $\{0,1\}$ sind **Wahrheitswerte**. (Intuition: 0 = falsch, 1 = wahr). Eine **Belegung** einer Menge $M = \{A, B, C, \ldots\}$ von atomaren Formeln ist eine Abbildung

$$\mathcal{A} : \{A, B, C, \ldots\} \to \{0,1\}$$

$\mathcal{A}$ wird auf alle Formeln erweitert durch

1. $\mathcal{A}(F \wedge G) = \min\{\mathcal{A}(F), \mathcal{A}(G)\} = \begin{cases} 1 & \text{if } \mathcal{A}(F) = \mathcal{A}(G) = 1 \\ 0 & \text{sonst} \end{cases}$
2. $\mathcal{A}(F \vee G) = \max\{\mathcal{A}(F), \mathcal{A}(G)\} = \begin{cases} 0 & \text{if } \mathcal{A}(F) = \mathcal{A}(G) = 0 \\ 1 & \text{sonst} \end{cases}$
3. $\mathcal{A}(\neg F) = 1 - \mathcal{A}(F) = \begin{cases} 0 & \text{if } \mathcal{A}(F) = 1 \\ 1 & \text{sonst} \end{cases}$

Beispiel 1.4. Man beachte, dass eine Formel laut Definition endlich ist. Also enthält jede Formel nur endlich viele atomare Formeln. Eine Belegung kann also als endlich Liste geschrieben werden, bzw. als Wertetabelle. Ist z.B. $\mathcal{A}(A) = 1, \mathcal{A}(B) = 1, \mathcal{A}(C) = 0$, so können wir das kurz schreiben als

A	B	C
1	1	0

Nun, was ist für dieses bestimmte $\mathcal{A}$, and for $F = \neg((A \wedge B) \vee C)$, was ist der Wert von $\mathcal{A}(F)$?

$$\mathcal{A}(\neg((A \wedge B) \vee C)) = 1 - \mathcal{A}((A \wedge B) \vee C) = 1 - \begin{cases} 0 & \text{if } \mathcal{A}(A \wedge B) = \mathcal{A}(C) = 0 \\ 1 & \text{sonst} \end{cases}$$
$$= 1 - \begin{cases} 0 & \text{if } \mathcal{A}(A \wedge B) = 0 \\ 1 & \text{sonst} \end{cases} \qquad = 1 - 1 = 0 \quad (\text{since } \mathcal{A}(A) = \mathcal{A}(B) = 1)$$

Natürlich ist dies eine sehr mühsame Methode. Eine bessere Methode liefern **Wahrheitswertetafeln**. Eine Formel mit n atomaren Formeln hat 2^n mögliche Belegungen $\mathcal{A}$. Daher können wir $\neg, \vee$ und $\wedge$ viel bequemer definieren durch by Wahrheitswertetafeln, und z.B. die Wahrheitswertetafeln dann auch erstellen für $\Rightarrow$ and $\Leftrightarrow$:

$\mathcal{A}(F)$	$\mathcal{A}(G)$	$\mathcal{A}(\neg(F))$	$\mathcal{A}(F \wedge G)$	$\mathcal{A}(F \vee G)$	$\mathcal{A}(F \Rightarrow G)$ $= \mathcal{A}(\neg F \vee G)$	$\mathcal{A}(F \Leftrightarrow G)$ $= \mathcal{A}((F \wedge G) \vee (\neg F \wedge \neg G))$
0	0	1	0	0	1	1
0	1		0	1	1	0
1	0	0	0	1	0	0
1	1		1	1	1	1

Die Zeilen der Wahrheitswertetafel entsprechen genau den möglichen Belegungen $\mathcal{A}$. Wir können nun auch $\mathcal{A}(F)$ aus Beispiel 1.4 ermitteln, indem wir die Wahrheitswertetafel für F aufstellen und in der richtigen Zeile (nämlich 1 1 0) das $\mathcal{A}(F)$ ablesen. Das ist nicht schwer (Übung).

Übliche Interpretationen dieser Regeln sind, dass $\vee$ soviel meint wie "oder", $\wedge$ soviel wie "und", und $\neg$ soviel wie "nicht". Bei und und nicht ist das in der Tat sehr passend. Bei "oder" könnte man streiten. Wenn jemand sagt, er wird Nudelsalat mitbringen oder Fleischbällchen, dann erwartet man ja nicht, dass er beides mitbringt, sondern nur eins von beiden. Genauso gilt: Wenn in einer Prüfung der Prüfer fragt, ob Antwort A richtig ist oder Antwort B richtig, so wird die Aussage "Ja" nicht gut ankommen. Obwohl sie bezüglich des logischen oder völlig korrekt ist (angenommen, dass mindestens eine der beiden Antworten A oder B richtig ist).

Aber erstens ist das ja nur eine Interpretation. Wir wollen hier ja gerade formalisieren, also kümmern wir uns nicht um die Interpretation.

Außerdem gibt es einen weiteren Operator, der die fehlende Bedeutung abdeckt: das "entweder oder", auch "exklusives oder", kurz oder $\oplus$: $\mathcal{A}(A \oplus B) = 1$ if and only if $\mathcal{A}(A) \neq \mathcal{A}(B)$. Es ist leicht zu sehen, dass $A \oplus B$ das Gleiche ist wie $\neg(A \Leftrightarrow B)$. (Übung: wie?)

Eine übliche Interpretation von $\Rightarrow$ ist die Implikation: "also", oder "aus A folgt B". Eine übliche Interpretation von $\Leftrightarrow$ ist die Äquivalenz: "genau dann, wenn". So werden die Zeichen ja in der Mathematik verwendet. Aber Obacht: wir müssen hier darauf achten, dass wir die Zeichen $\Rightarrow$ und $\Leftrightarrow$ nicht mit ihrer Metabedeutung vermischen: Es würden die Ebenen vermischt, wenn wir schreiben $\mathcal{A}(A \Rightarrow B) = 0 \Rightarrow \mathcal{A}(A \Leftrightarrow B) = 0$. Wir werden das hier also sorgfältig unterscheiden, und z.B. stattdessen schreiben $\mathcal{A}(A \Rightarrow B) = 0$, also $\mathcal{A}(A \Leftrightarrow B) = 0$.

Um Klammern zu sparen führen wir eine Hierarchie zwischen den Operatoren ein (analog zu Punkt-vor-Strich-Rechnung, also "$\cdot$ vor $+$ und $-$):

$\neg$ vor $\wedge$ und $\vee$ vor $\Rightarrow$ vor $\Leftrightarrow$.

Also schreiben wir z.B. nur $\neg A \vee B \Rightarrow C$ statt $((\neg A) \vee B) \Rightarrow C$.

Die **Hauptfragen**, denen wir uns im Folgenden widmen, sind:

1. Welche Formeln haben die gleiche Bedeutung? (**Äquivalenz**)

2. Gegeben eine Formel F, gibt es eine Belegung, die sie wahr macht? (**Erfüllbarkeit**)

3. Gegeben zwei Formeln F, G, ist G eine logische Folgerung von F? D.h., wann immer F wahr ist, ist auch G wahr? (**logische Folgerung**)

Neben dem präzisen formalen Aufbau liegt der Fokus dieser Vorlesung auf dem algorithmischen

Beantworten dieser Fragen. Dabei heißt "algorithmisch" hier, die grundlegenden (Meta-)Algorithmen kennenzulernen und zu verstehen. Wir werden hier nichts implementieren oder Probleme mit SAT-Solvern lösen.

Für die Fragen oben definieren wir jetzt die benötigten Begriffe.

Bemerkung 1.5. Wenn wir im Folgenden etwa $\mathcal{A}(F)$, oder $\mathcal{A}(F) = \mathcal{A}(G)$, usw. schreiben, dann gehen wir immer davon aus, dass $\mathcal{A}$ für alle atomaren Formeln in F — bzw. in F und G, usw. — definiert ist. Das erspart uns eine weiter technische Definition.

Definition 1.6. Eine Belegung $\mathcal{A}$ **erfüllt** eine Formel F falls $\mathcal{A}(F) = 1$. Notation: $\mathcal{A} \models F$.

Sonst ($\mathcal{A}(F) = 0$) sagen wir, $\mathcal{A}$ erfüllt F nicht. Notation: $\mathcal{A} \not\models F$.

F ist **erfüllbar**, falls es eine Belegung $\mathcal{A}$ gibt, so dass $\mathcal{A}(F) = 1$. Sonst heißt F unerfüllbar.

F ist eine **Tautologie**, falls für alle Belegungen $\mathcal{A}$ gilt: $\mathcal{A}(F) = 1$.

F ist **äquivalent** zu G falls für alle Belegungen $\mathcal{A}$ gilt: $\mathcal{A}(F) = \mathcal{A}(G)$. Notation: $F \equiv G$.

G ist eine **logische Folgerung** von F, falls für alle Belegungen $\mathcal{A}$ mit $\mathcal{A}(F) = 1$ gilt, dass auch $\mathcal{A}(G) = 1$. Notation: $F \models G$.

Beispiel 1.7. $A \vee \neg A$ und $\neg A \Rightarrow (A \Rightarrow B)$ sind Tautologien. $A \wedge \neg A$ ist unerfüllbar (Wahrheitswertetafeln!)

Bemerkung 1.8. Im Allgemeinen könnten F und G komplett verschiedene atomare Formeln enthalten und dennoch kann $F \equiv G$ gelten. (Etwa falls beide Formeln Tautologien, z.B. $F = A \vee \neg A$ and $G = \neg(B \wedge \neg B)$).

1.2 Rechenregeln

Rechenregeln wie $F \wedge G \equiv G \wedge F$ sind offensichtlich (oder?). Aber es gibt weniger offensichtliche.
18. Okt. Hier die wichtigsten (plus einige weniger wichtige).

Satz 1.9. *Seien F, G, H Formeln. Dann gelten die folgenden Äquivalenzen:*

$F \wedge F \equiv F$, $F \vee F \equiv F$	*(Idempotenz)*
$F \wedge G \equiv G \wedge F$, $F \vee G \equiv G \vee F$	*(Kommutativität)*
$(F \wedge G) \wedge H \equiv F \wedge (G \wedge H)$, $(F \vee G) \vee H \equiv F \vee (G \vee H)$	*(Assoziativität)*
$F \wedge (F \vee G) \equiv F$, $F \vee (F \wedge G) \equiv F$	*(Absorption)*
$F \wedge (G \vee H) \equiv (F \wedge G) \vee (F \wedge H)$, $F \vee (G \wedge H) \equiv (F \vee G) \wedge (F \vee H)$	*(Distributivität)*
$\neg\neg F \equiv F$	
$\neg(F \wedge G) \equiv \neg F \vee \neg G$, $\neg(F \vee G) \equiv \neg F \wedge \neg G$	*(de Morgansche Regeln)*
Ist F eine Tautologie, dann gilt $F \vee G \equiv F$, $F \wedge G \equiv G$	
Ist F unerfüllbar, dann gilt $F \vee G \equiv G$, $F \wedge G \equiv F$	

Alle Äquivalenzen können mit Wahrheitswertetafeln gezeigt werden, siehe auch Übungsblatt 2.

Wegen der Assoziativität können wir kurz schreiben $F \wedge G \wedge H$ statt $(F \wedge G) \wedge H$ oder $F \wedge (G \wedge H)$.

Beispiel 1.10. Wir können natürlich Wahrheitswertetafeln nutzen, um die Äquivalenz irgendwelcher Formeln zu zeigen. Nun, da wir die Regeln oben zur Verfügung haben (wir tun mal so, als hätten wir die alle bewiesen), konnen wir die Äquivalenz zweier Formeln, z.B. $F = (A \vee (B \vee C)) \wedge (C \vee \neg A)$ und

$G = (\neg A \wedge B) \vee C$ auch mit den Rechenregeln oben zeigen:

$$(A \vee (B \vee C)) \wedge (C \vee \neg A) \overset{(Ass.,Komm.)}{\equiv} (C \vee (A \vee B)) \wedge (C \vee \neg A) \overset{(Distr.)}{\equiv} C \vee ((A \vee B) \wedge \neg A)$$
$$\overset{(Komm.,Distr)}{\equiv} C \vee ((\neg A \wedge A) \vee (\neg A \wedge B)) \overset{(unsat.)}{\equiv} C \vee (\neg A \wedge B) \overset{(Komm.)}{\equiv} (\neg A \wedge B) \vee C.$$

Wenn man sehr genau ist, könnte man nun einwenden, dass wir ja z.B. wissen, dass für die Teilformeln oben z.B. gilt $A \vee (B \vee C) \equiv C \vee (A \vee B)$ gilt. Aber wieso gilt dann auch, dass die kompletten Formeln (die diese Teilformeln enthalten), auch äquivalent sind? Tatsächlich muss das bewiesen werden.

Satz 1.11 (Ersetzungssatz)**.** *Seien F, G Formeln mit $F \equiv G$. Sei F eine Teilformel von H. Sei H' die Formel, die aus H entsteht, wenn F durch G ersetzt wird. Dann gilt $H \equiv H'$.*

Proof. (Mittels struktureller Induktion, über den induktiven Aufbau einer Formel)

Induktionsanfang: Sei H eine atomare Formel. Dann hat H nur eine einzige Teilformel, nämlich H selbst: Also $H = F$, also $H' = G$, hence $H' = G \equiv F = H$.

Induktionsschritt: Sei F eine Formel. Sei die Behauptung wahr für alle Formeln, die kleiner sind als F (insbesondere also für alle Teilformeln von F außer F selbst).

Fall 0: $H = F$, dann genau wie oben.

Fall 1: $H = \neg H_1$. Wegen der Induktionsannahme ist $H_1 \equiv H_1'$, also $H = \neg H_1 \equiv \neg H_1' = H'$. Eigentlich noch genauer: für jedes $\mathcal{A}$ gilt $\mathcal{A}(H) = 1 - \mathcal{A}(H_1) = 1 - \mathcal{A}(H_1') = \mathcal{A}(\neg H_1') = \mathcal{A}(H')$.

Fall 2: $H = H_1 \vee H_2$. Nehmen wir an, dass (OBdA[1]) gilt, dass F Teilformel von H_1 ist. (Sonst benennen wir sie um, H_2 wird dann zu H_1). Wegen der Induktionsannahme ist $H_1 \equiv H_1'$, also $H = H_1 \vee H_2 \equiv H_1' \vee H_2 = H'$.

Fall 3: $H = H_1 \wedge H_2$, das geht komplett analog zu Fall 2. □

An diesem Beweis wird aber auch klar, dass viele Beweise in der Formalen Logik technisch sind, wenig elegant, und dass man nicht sehr viel daraus lernen kann. (Außer formales beweisen.) Im Folgenden nehmen wir uns also die Freiheit, nur die lehrreichen oder schönen Beweise zu zeigen, und die technischen wegzulassen (die meisten stehen in dem Buch von Schöning).

1.3 Normalformen

Wahrheitswertetafeln sind eine erste algorithmische Methode, um die Frage nach Erfüllbarkeit einer Formel F zu beantworten (wie?), oder ob zwei Formeln F und G äquivalent sind (wie?). Das ist aber sicher nicht sehr effizient: Enthält F etwa n atomare Formeln, dann ist die Anzahl der Zeilen in der zugehörigen Wahrheitswertetafel 2^n. Im Folgenden lernen wir drei effizientere Verfahren kennen, um die Frage "ist F erfüllbar?" effizient zu beantworten, in einem Fall nur für eine gewisse Klasse von Formeln ("Hornformeln"). Für die ersten beiden Verfahren (Hornformelalgorithmus, Resolutionskalkül) müssen die Formeln eine bestimmte Form haben, eine "Normalform". **Notation:** $\bigwedge_{i=1}^{n} F_i = F_1 \wedge F_2 \wedge \cdots \wedge F_n$; $\bigvee_{i=1}^{n} F_i = F_1 \vee F_2 \vee \cdots \vee F_n$.

[1] "Ohne Beschränkung der Allgemeinheit"

Definition 1.12. Ein **Literal** ist eine Formel der Gestalt A or $\neg A$, wobei A eine atomare Formel ist.

Eine Formel F hat **disjunktive Normalform (DNF)**, falls

$$F = \bigvee_{i=1}^{n} \left(\bigwedge_{j=1}^{m_i} L_{ij} \right) \quad \text{wobei die } L_{ij} \text{ Literale sind.}$$

Eine Formel der Form $L_1 \wedge L_2 \wedge \cdots \wedge L_n$ (wobei die L_i Literale sind) heißt *Konjunktionsklausel*.

Eine Formel F hat **konjunktive Normalform (KNF)**, falls

$$F = \bigwedge_{i=1}^{n} \left(\bigvee_{j=1}^{m_i} L_{ij} \right) \quad \text{wobei die } L_{ij} \text{ Literale sind.}$$

Eine Formel der Form $L_1 \vee L_2 \vee \cdots \vee L_n$ (wobei die L_i Literale sind) heißt *Disjunktionsklausel*.

Beispiel 1.13. Die folgenden Formeln sind alle in DNF:

$$(A \wedge \neg B \wedge \neg C) \vee (\neg D \wedge E \wedge F), \quad (A \wedge B) \vee C, \quad A \wedge B, \quad A.$$

Die folgenden Formeln sind nicht in DNF:

$\neg(A \vee B)$ (weil ein "oder" innerhalb eines "nicht" sitzt), $A \vee (B \wedge (C \vee D))$ weil ein "oder" innerhalb eines "und" sitzt).

Satz 1.14. *Jede Formel hat eine äquivalente Formel in DNF, und auch eine äquivalente Formel in KNF.*

Der Beweis dieses Satzes ist lang und technisch. Er benutzt wieder strukturelle Induktion, und besteht im Wesentlichen darin, zu zeigen, dass der folgende Algorithmus terminiert und immer eine KNF liefert.

Algorithmus 1.15 (KNF). Ersetze alle Teilformeln von F der Form $G \Rightarrow H$ durch $\neg G \vee H$, und alle Teilformeln der Form $G \Leftrightarrow H$ durch $(G \wedge H) \vee (\neg G \wedge \neg H)$.

1. Ersetze in F jedes...
   - $\neg\neg G$ durch G
   - $\neg(G \wedge H)$ durch $\neg G \vee \neg H$
   - $\neg(G \vee H)$ durch $\neg G \wedge \neg H$, solange wie möglich. Dann
2. Ersetze in F jedes...
   - $G \vee (H \wedge J)$ durch $(G \vee H) \wedge (G \vee J)$
   - $(G \wedge H) \vee J$ durch $(G \vee J) \wedge (H \vee J)$, solange wie möglich. Dann
3. Streiche Wiederholungen von Klauseln, falls nötig.

Der entsprechende Algorithmus zur Konstruktion der DNF einer Formel F sieht fast genauso aus. Es wird nur Schritt 2 oben ersetzt durch:

2. Ersetze in F jedes...

- $G \wedge (H \vee J)$ durch $(G \wedge H) \vee (G \wedge J)$

- $(G \vee H) \wedge J$ durch $(G \wedge J) \vee (H \wedge J)$, solange wie möglich. Dann...

Beispiel 1.16. Sei $F = (A \wedge B \wedge C) \vee (D \wedge E)$. Das ist eine DNF. Wir nutzen nun den KNF-Algorithmus, um F in KNF umzuformen:

$$\begin{aligned}(A \wedge B \wedge C) \vee (D \wedge E) &\equiv \big((A \wedge B \wedge C) \vee D\big) \wedge \big((A \wedge B \wedge C) \vee E\big)\\ &\equiv (A \vee D) \wedge (B \vee D) \wedge (C \vee D) \wedge (A \vee E) \wedge (B \vee E) \wedge (C \vee E)\end{aligned}$$

Bemerkung 1.17. Hier haben wir strenggenommen eine Verallgemeinerung des Distributivitätsgesetzes benutzt. Rechnet man ein paar Beispiele ausführlich, sieht man schnell, dass das Distributivitätsgesetz aus Satz 1.9 sich auf drei, vier ... Teilformeln verallgemeinert. (Im Beispiel oben nutzten wir "drei"). Die allgemeinste Form der Distributivitätsgesetze sieht so aus:

$$(\bigvee_{i=1}^{n} F_i) \wedge (\bigvee_{j=1}^{m} G_j) \equiv \bigvee_{i=1}^{n} (\bigvee_{j=1}^{m} (F_i \wedge G_j)) \quad \text{resp.} \quad (\bigwedge_{i=1}^{n} F_i) \vee (\bigwedge_{j=1}^{m} G_j) \equiv \bigwedge_{i=1}^{n} (\bigwedge_{j=1}^{m} (F_i \vee G_j))$$

Nebenbei bemerkt gibt es einen weiteren Algorithmus zum Erstellen der KNF oder DNF einer Formel. Der nutzt Wahrheitswertetafeln. Angenommen, wir haben bereits die Wahrheitswertetafel einer Formel F, wobei F die atomaren Formeln $A_1, \ldots, A_n$ enthält.

Algorithmus 1.18. Um die DNF von F zu erstellen:

- Jede Reihe mit $\mathcal{A}(F) = 1$ liefert eine Konjunktionsklausel $(L_1 \wedge \cdots \wedge L_n)$. Falls in dieser Zeile $\mathcal{A}(A_i) = 1$, dann setze $L_i = A_i$, sonst $L_i = \neg A_i$ $(i = 1, \ldots, n)$
- Verbinde die Klauseln aus dem letzten Schritt mit $\vee$.

Um die KNF von F zu erstellen:

- Jede Reihe mit $\mathcal{A}(F) = 0$ liefert eine Konjunktionsklausel $(L_1 \vee \cdots \vee L_n)$. Falls in dieser Zeile $\mathcal{A}(A_i) = 0$, dann setze $L_i = A_i$, sonst $L_i = \neg A_i$ $(i = 1, \ldots, n)$
- Verbinde die Klauseln aus dem letzten Schritt mit $\wedge$.

Beispiel 1.19. Betrachte $F = \neg(A \Rightarrow B) \vee \neg(A \vee B \vee C)$. Die Wahrheitswertetafel dazu ist

A	B	C	F
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

Wir lesen für die DNF ab:

$$F \equiv (\neg A \wedge \neg B \wedge \neg C) \vee (A \wedge \neg B \wedge \neg C) \vee (A \wedge \neg B \wedge C).$$

Wir lesen für die KNF ab:

$$F \equiv (A \vee B \vee \neg C) \wedge (A \vee \neg B \vee C) \wedge (A \vee \neg B \vee \neg C) \wedge (\neg A \vee \neg B \vee C) \wedge (\neg A \vee \neg B \vee \neg C)$$

Im Allgemeinen sind KNF und DNF einer Formel nicht eindeutig, es gibt meistens mehrer. Der letzte Algorithmus liefert allerdings eine Art Standard-KNF bzw Standard-DNF, in der jede Klausel alle atomaren Formeln enthält. Bis auf die Reihenfolge ist die eindeutig.

Zusammenhang mit P vs NP

25. Okt.

Bemerkung 1.20. Die Frage "ist eine aussagenlogische Formel F erfüllbar" ist NP-vollständig (Cook's Theorem 1971, siehe auch Levin 1973). Cook zeigt, dass die Frage nach der Erfüllbarkeit einer allgemeinen Formel (das Problem heißt SAT) sich in polynomieller Zeit zurückführen lässt auf die Erfüllbarkeit einer Formel in KNF (das Problem heißt dann manchmal CNFSAT). Dieses letztere Problem lässt sich wiederum reduzieren auf die Erfüllbarkeit einer Formel in KNF, in der jede Klausel höchstens drei Literale hat. Diese Problem ist berühmt und hat den Namen 3SAT ("three-satisfiability"). 3SAT ist also auch NP-collständig (Karp 1972). Im nächsten Kapitel lernen wir eine Klasse von Formeln kennen (sogenannte "Hornformeln"), für die die Frage nach der Erfüllbarkeit effizient entscheidbar ist. Das entsprechende Problem 2SAT — Erfüllbarkeit einer Formel in KNF, in der jede Klausel höchstens zwei Literale hat — ist in polynomieller Zeit entscheidbar.

Bemerkung 1.21. Das entsprechende Problem der Lösung einer Formel in DNF ist entscheidbar in polynomieller Zeit. Daher können wir nicht erwarten, dass es einen allgemeinen *und* effizienten Weg gibt, eine beliebige Formel in KNF in eine äquivalente Formel in DNF umzuformen (mit polynomiellen Kosten). Tatsächlich kann bereits die Länge der Formel im Allgemeinen exponentiell wachsen. Dieser schlimmste Fall wird beispielsweise erreicht durch

$$F = \bigwedge_{i=1}^{n} A_i \vee B_i = (A_1 \vee B_1) \wedge (A_2 \vee B_2) \wedge \cdots \wedge (A_{n-1} \vee B_{n-1}) \wedge (A_n \vee B_n)$$

(diese KNF hat größenordnungsmäßig Länge $2n$). Die DNF von F ist

$$\begin{array}{r} (A_1 \wedge A_2 \wedge \cdots \wedge A_{n-2} \wedge A_{n-1} \wedge A_n) \\ \vee (A_1 \wedge A_2 \wedge \cdots \wedge A_{n-2} \wedge A_{n-1} \wedge B_n) \\ \vee (A_1 \wedge A_2 \wedge \cdots \wedge A_{n-2} \wedge B_{n-1} \wedge A_n) \\ \quad \vdots \quad \vdots \\ \vee (B_1 \wedge B_2 \wedge \cdots \wedge B_{n-2} \wedge B_{n-1} \wedge A_n) \\ \vee (B_1 \wedge B_2 \wedge \cdots \wedge B_{n-2} \wedge B_{n-1} \wedge B_n) \end{array},$$

mit einer Länge von 2^n. (Diese Längenexplosion passiert auch andersrum, von DNF zu KNF, wenn wir alle $\vee$ durch $\wedge$ ersetzen und umgekehrt.)

SAT is NP-complete	(Cook 1971, Levin 1973)
KNFSAT is NP-complete	(Cook 1971)
3SAT is in NP-complete	(Karp 1972)
2SAT is in P	(exercise)
DNFSAT is in P	(exercise)

Table 1: Übersicht über die Schwierigkeit der Erfüllbarkeitsprobleme (zur Erläuterung von P und NP siehe "Komplexität" auf Seite 58)

Satz 1.22 ((3SAT))**.** *Für jede Formel F gibt es eine Formel G in KNF, in der jede Klausel höchstens drei Literale hat, so dass F genau dann erfüllbar, ist wenn G erfüllbar ist.*

Proof. Wegen Satz 1.14 hat F eine KNF. Wir müssen nur zeigen, wie man jede Disjunktionsklausel mit mehr als drei Literalen ersetzt durch eine oder mehrere Disjunktionsklauseln mit höchstens drei Literalen. Der Trick besteht darin, jede Klausel mit m Literalen zu ersetzen durch vier Klauseln mit $m-1, 3, 2$- und 2-Literalen.

Angenommen, es gibt eine Klausel H mit mehr als drei Literalen: $H = A \vee B \vee G$. Wir ersetzen H in die KNF durch $H' = (A' \Leftrightarrow A \vee B) \wedge (A' \vee G)$. Dabei führen wir eine neue atomare Formel A' ein, die in F noch nicht vorkommt. Wenn H erfüllbar ist, kann jede Belegung $\mathcal{A}$ für H mit $\mathcal{A}(H) = 1$ erfolgen zu H' erweitert werden, indem $\mathcal{A}(A') := \mathcal{A}(A \vee B)$ gesetzt wird.

Warum hilft das? Wir behaupten, dass (a) $\mathcal{A}(H) = \mathcal{A}(H')$ und (b) H' kann als Disjunktionsklausel mit $m-1, 3, 2$ und 2 Literalen geschrieben werden.

Zu (a) Für das obige $\mathcal{A}$ gilt

$$\mathcal{A}(H) = \mathcal{A}(A \vee B \vee G) = \mathcal{A}(A' \vee G) = \mathcal{A}\big((A' \Leftrightarrow A \vee B) \wedge (A' \vee G)\big) = \mathcal{A}(H').$$

Die vorletzte Gleichung gilt, weil $\mathcal{A}(A' \Leftrightarrow A \vee B) = 1$ (da wir definert haben $\mathcal{A}(A') = \mathcal{A}(A \vee B)$, vergleiche die Tautologieregel im Satz 1.9. Somit ist H' genau dann erfüllbar, wenn H es ist. (In eine Richtung: da jedes $\mathcal{A}$ für H im anderen auf $\mathcal{A}$ für H' erweitert werden kann, in der anderen Richtung, da die Einschränkung der größeren Belegung (für H') die ursprüngliche Belegung $\mathcal{A}$ (für H) ergibt durch Ignorieren von A').

(b) Das ist nun einfach:

$$\begin{aligned} A' \Leftrightarrow A \vee B &\equiv (A' \Rightarrow A \vee B) \wedge (A \vee B \Rightarrow A') \equiv (\neg A' \vee A \vee B) \wedge (\neg(A \vee B) \vee A') \\ &\equiv (\neg A' \vee A \vee B) \wedge ((\neg A \wedge \neg B) \vee A') \equiv (\neg A' \vee A \vee B) \wedge (\neg A \vee A') \wedge (\neg B \vee A'), \end{aligned}$$

also

$$H' \equiv (\neg A' \vee A \vee B) \wedge (\neg A \vee A') \wedge (\neg B \vee A') \wedge (A' \vee G),$$

also hat die neue Formel auch KNF. Außerdem folgt die Behauptung über die Anzahl der Klauseln. □

Zählen wir auch, wie die Anzahl der Klauseln dabei insgesamt wächst. Das Ersetzen von H durch H' bedeutet, dass eine einzelne Klausel mit $m \geq 4$ Literalen ersetzt wird durch eine Klausel mit $m-1$-Literalen plus eine weitere Klausel mit drei Literalen plus zwei Klauseln mit zwei Literalen. Dies tun wir solange, bis alle Klauseln drei Literale oder weniger haben. Falls also unsere Formel F in KNF n Klauseln hat mit jeweils höchstens m Literalen, dann können wir jede Klausel in $m-3$ Schritten von reduzieren auf mehrere kleine Klauseln mit höchstens drei Literalen. Wie viele kleine Klauseln? In jedem Schritt tauschen wir eine Klausel gegen vier Klauseln, also wächst in jedem Schritt die Zahl um drei. Daher wird jede lange Klausel (mit der Länge m) ersetzt durch höchstens $3(m-3)$ der Länge drei (oder weniger). Alles in allem haben wir also höchstens $3n(m-3)$ Klauseln. Dies ist ein Polynom in n, bzw. in m, bzw in $\max\{n, m\}$.

Nun lernen wir eine Klassen von Formeln kennen, für die es ein effizientes Entscheidungsverfahren zur Erfüllbarkeit gibt.

1.4 Hornformeln

Definition 1.23. Eine Formel F heißt *Hornformel*, wenn sie in KNF und vorliegt wenn jede Disjunktonsklausel höchstens ein positives Literal hat (d.h., ein Literal, das kein $\neg$ enthält).

Zum Beispiel ist $F = (\neg A \vee \neg B \vee C) \wedge (A \vee \neg D) \wedge (\neg A \vee \neg B \vee \neg C) \wedge D \wedge \neg E$ eine Hornformel, wohingegen $G = (A \vee B \vee \neg C) \wedge (A \vee \neg B \vee \neg C)$ keine Hornformel ist.

Jede Hornformel kann mithilfe von Implikationen intuitiver geschrieben werden. Beispielsweise kann die obige Formel F wie folgt geschrieben werden:

$$(A \wedge B \Rightarrow C) \wedge (D \Rightarrow A) \wedge (A \wedge B \wedge C \Rightarrow 0) \wedge (1 \Rightarrow D) \wedge (E \Rightarrow 0),$$

wobei 1 für eine beliebige Tautologie steht, und 0 für beliebige unerfüllbare Formel.

Hier der versprochene effiziente Algorithmus zur Überprüfung der Erfüllbarkeit einer Hornformel.

Algorithmus 1.24. Input: eine Hornformel F.

1. If F enthält eine Teilformel $(1 \Rightarrow A_i)$, dann markiere alle (!) Literale in F, die A_i enthalten.
2. **while** F enthält eine Teilformel G mit $G = (A_1 \wedge \cdots \wedge A_n \Rightarrow B)$ $(n \geq 1)$ wobei alle A_i markiert sind (und B nicht) **do**

   if $(B = 0)$ return "F ist unerfüllbar" STOP

   sonst $B \neq 0$ (i.e. B ist Literal) **then** markiere alle Literale in F, die B enthalten.
3. Return "F ist erfüllbar" STOP

Eine Belegung, die F erfüllt, ist dann gegeben, indem man $\mathcal{A}(A_i) = 1$ setzt für alle markierten atomaren Formeln A_i und $\mathcal{A}(A_j) = 0$ für alle unmarkierten atomaren Formeln A_j.

Satz 1.25. *Der obige Algorithmus ist für alle Hornformeln korrekt. Er stoppt nach höchstens k Markierungsschritten, wobei k die Anzahl der verschiedenen atomaren Formeln in F bezeichnet. Darüber hinaus liefert der Algorithmus eine* kleinste *Belegung $\mathcal{A}$, die F erfüllt. Das heißt, für jede andere Belegung $\mathcal{A}'$ mit $\mathcal{A}' \models F$ gilt, dass aus $\mathcal{A}(A_i) = 1$ folgt, dass $\mathcal{A}'(A_i) = 1$ gilt.*

Die kleinste Belegung aus der letzten Aussage ist eindeutig. Das kann wie folgt gezeigt werden: wenn der Algorithmus startet sind alle atomaren Formeln A_i unmarkiert, d.h. noch ist $\mathcal{A}(A_i) = 0$. Alle Markierungen sind erzwungen, d.h. alle markierten A_i müssen zwingend den Wert 1 haben. Wenn F erfüllbar ist, stoppt der Algorithmus mit einer erfüllenden Belegung, in der alle atomaren Formeln mit dem Wert 1 gezwungen sind, den Wert 1 zu haben. In diesem Sinne erhalten wir eine minimale Belegung $\mathcal{A}$.

1.5 Kompaktheitssatz

31. Okt.

Statt einzelner Formeln können wir auch nach der Erfüllbarkeit von mehreren Formeln (endlich viele oder unendlich viele) auf einmal fragen. Das heißt, ist $M = \{F_1, F_2, \ldots, F_n\}$ eine Menge von Formeln, dann wollen wir eine Belegung $\mathcal{A}$ finden, so dass $\mathcal{A}(F_i) = 1$ für alle $i = 1, \ldots, n$. Für endlich viele Formeln liefert das nichts wirklich neues.

Bemerkung 1.26. Eine Menge von Formeln $\{F_1, F_2, \ldots, F_n\}$ ist erfüllbar genau dann, wenn $F = \bigwedge_{i=1}^{n} F_i$ erfüllbar ist.

Interessant — und für spätere Zwecke nötig — wird es, wenn wir unendliche Mengen von Formeln betrachten. In diesem Zusammenhang ist der nächste Satz zentral.

Satz 1.27 (Kompaktheitssatz)**.** *Eine unendliche Menge M von Formeln ist erfüllbar genau dann, wenn jede endliche Teilmenge von M erfüllbar ist.*

Der exakte Beweis ist lang und technisch, siehe z.B. UWE SCHÖNING: LOGIK FÜR INFORMATIKER. Eine Beweisskizze steht weiter unten. Aber der Beweis nutzt ein interessantes Resultat:

Satz 1.28 (Königs Lemma)**.** *Sei T ein Baum mit unendlich vielen Knoten, so dass jeder Knoten nur endlich viele Nachbarn hat. Dann enthält T einen unendlichen Pfad. (Das ist ein Pfad $v_1 v_2 \cdots$ mit unendlich vielen Knoten, so dass $v_i \neq v_j$ für $i \neq j$.)*

Proof. (Komplett aus Wikipedia:) Sei v_i die Menge der Knoten. Da T ein unendlicher Baum ist, wissen wir, dass diese Knotenmenge unendlich ist und der Graph zusammenhängend (d. h. für je zwei Knoten v_i, v_j gibt es in T einen Pfad von v_i nach v_j).

Beginnen wir mit einem beliebigen Knoten v_1. Jeder der unendlich vielen Knoten von G kann von v_1 mit einem (einfachen) Pfad erreicht werden, und jeder dieser Pfade muss mit einem der endlich vielen Knoten beginnen, die Nachbarn von v_1 sind. Über mindestens einen dieser Nachbarn konnen unendlich viele Knoten erreicht werden, ohne über v_1 zu gehen. (Wenn nicht, dann wäre der gesamte Graph die Vereinigung endlich vieler endlicher Mengen, also endlich.) Wählen wir also einen dieser Knoten aus und nennen ihn v_2.

Nun können von v_2 aus unendlich viele Knoten von T erreicht werden, ohne den Knoten v_1 zu benutzen. Jeder dieser Pfade muss beginnen mit einem der endlich Nachbarknoten von v_2. Wieder, analog zum Argument oben, müssen über mindestens einen dieser Nachbarknoten unendlich viele Knoten erreicht werden. Wählen wir einen dieser Nachbarknoten aus und nennen wir ihn v_3. Weitermachen. □

Aus dem Lemma von König folgt der Kompaktheitssatz im Wesentlichen so:

Zeichne einen Baum. Ein Knoten ist die Wurzel. Seine Kinder sind alle erfüllenden Belegungen $\mathcal{A}_1, \ldots, \mathcal{A}_k$ für F_1. Die Kinder von $\mathcal{A}_i$ sind alle erfüllenden Belegungen $\mathcal{A}'_1, \ldots, \mathcal{A}'_m$ für $F_1 \wedge F_2$. mit $\mathcal{A}'_j(A_\ell) = \mathcal{A}_i(A_\ell)$ für alle atomaren Formeln A_ℓ in F_1. (Also ist $\mathcal{A}'_j$ eine Erweiterung von $\mathcal{A}_i$ auf diejenigen atomaren Formeln in F_2, die in F_1 noch nicht vorkamen).

Wir wiederholen dasselbe für jedes $\mathcal{A}'_j$: Seine Knoten sind alle Erweiterungen von $\mathcal{A}'_j$ zu allen atomaren Formeln in F_3, die $F_1 \wedge F_2 \wedge F_3$ erfüllen. Aufgrund der Bedingung des Kompaktheitssatzes gibt es unendlich viele erfüllende Belegungen (entsprechend den Knoten im Baum). Durch die Konstruktion hat jeder Knoten nur endlich viele Kinder (möglicherweise keine), daher endlich viele Nachbarn. Jetzt folgt mit Königs Lemma, dass es einen beliebig langen Pfad in dem so konstruierten Baum gibt. Dieser Weg entspricht der Belegung, die alle Formeln erfüllt.

Beachten Sie, dass der Beweis uns nicht sagt, welche Zahlen ausreichen: Er zeigt es nur die Existenz einer solchen Sequenz, nicht die Konstruktion.

1.6 Logische Folgerungen I

Definition 1.29. Eine Formel G ist eine **Folgerung** (oder *logische Folgerung*, englisch "consequence") der Formeln $F_1, \ldots, F_n$, falls gilt: aus $\mathcal{A}(F_1) = \cdots = \mathcal{A}(F_n) = 1$ folgt $\mathcal{A}(G) = 1$. In diesem Fall schreiben wir $\{F_1, \ldots, F_n\} \models G$ (vgl. Definition 1.6).

Der Unterschied zwischen "impliziert" ("$\Rightarrow$") und "ist Konsequenz von" ("$\models$") ist subtil. Implikation ist rein auf der syntaktischen Ebene. $F \Rightarrow G$ bedeutet einfach $\neg F \vee G$. Die Konsequenz liegt auf einer konkreteren Ebene: für jede Belegung, die F wahr macht, gilt, dass unter ihr auch G wahr ist. In anderen Worten: $F \models G$ bedeutet, dass aus $\mathcal{A} \models F$ folgt, dass $\mathcal{A} \models G$ (für alle möglichen $\mathcal{A}$). Darüber hinaus kann die folgende Bemerkung (und ihr Beweis) hilfreich sein.

Lemma 1.30. *Die folgenden Aussagen sind äquivalent:*

1. $\{F_1, \ldots, F_n\} \models G$,
2. $F_1 \wedge \cdots \wedge F_n \Rightarrow G$ *ist eine Tautology,*
3. $F_1 \wedge \cdots \wedge F_n \wedge \neg G$ *ist unerfüllbar.*

Proof. Für die Äquivalenz von 1 und 2 zeigen wir die beiden Implikationen separat.

1. impliziert 2.: Fall 1: Sei $\mathcal{A}(F_1) = \cdots = \mathcal{A}(F_n) = 1$. Da G eine Konsequenz von F ist, impliziert dies $\mathcal{A}(G) = 1$. In diesem Fall ist $F_1 \wedge \cdots \wedge F_n \Rightarrow G$ wahr.

Fall 2: Es seien nicht alle $\mathcal{A}(F_i)$ gleich eins. Dann ist $\mathcal{A}(F_1 \wedge \cdots \wedge F_n) = 0$, daher ist $F_1 \wedge \cdots \wedge F_n \Rightarrow G$ wahr. In beiden Fällen ist also $F_1 \wedge \cdots \wedge F_n \Rightarrow G$ wahr es ist eine Tautologie.

2. impliziert 1.: Sei $H = F_1 \wedge \cdots \wedge F_n \Rightarrow G$ eine Tautologie.
Fall 1: Es gibt $\mathcal{A}$ mit $\mathcal{A}(F_i) = 0$. Dann $\mathcal{A}(F_1 \wedge \cdots \wedge F_n) = 0$, daher $\mathcal{A}(H) = 1$.

Fall 2: $\mathcal{A}(F_1) = \cdots = \mathcal{A}(F_n) = 1$. Weil $\mathcal{A}(H) = 1$ ist, folgt $\mathcal{A}(G) = 1$. Somit ist in beiden Fällen $\{F_1, \ldots, F_n\} \models G$.

Für die Äquivalenz von 2. und 3. müssen wir vergleichen, ob $F_1 \wedge \cdots \wedge F_n \Rightarrow G$ ist eine Tautologie mit ob $F_1 \wedge \cdots \wedge F_n \wedge \neg G$ ist unerfüllbar.

Sei $F_1 \wedge \cdots \wedge F_n \Rightarrow G$ eine Tautologie. Das heisst dass $\neg(F_1 \wedge \cdots \wedge F_n \Rightarrow G)$ unerfüllbar ist. Wegen

$$\neg(F_1 \wedge \cdots \wedge F_n \Rightarrow G) \equiv \neg(\neg(F_1 \wedge \cdots \wedge F_n) \vee G) \equiv (F_1 \wedge \cdots \wedge F_n) \wedge \neg G$$

auch letzterer Begriff ist unerfüllbar. □

Es wäre verlockend, Wahrheitswerttafeln zu verwenden. Aber das funktioniert hier nicht wirklich. Wenn wir auf der Anwendung von Wahrheitswerttafeln bestehen würden: Wir müssen dabei beachten, dass folgende Situation nicht passieren kann:

$$\mathcal{A}(F_1) = \cdots = \mathcal{A}(F_n) = 1 \text{ und } \mathcal{A}(G) = 0.$$

In der einen Richtung, weil G eine Folge von $\{F_1, \ldots, F_n\}$ ist. In der anderen Richtung, weil $F_1 \wedge \cdots \wedge F_n \Rightarrow G$ eine Tautologie ist. Daher müssten wir nur bestimmte Zeilen der Wahrheitswertetafel berücksichtigen.

$\{F, F \Rightarrow G\} \models G$	*(modus ponens)*
$\{\neg G, F \Rightarrow G\} \models \neg F$	*(modus tollens)*
$\{F \vee G, \neg F \vee H\} \models G \vee H$	*(resolution)*
$\{F, F \Rightarrow G\} \models G$	*(modus ponendo tollens)*
$\{F \Rightarrow G\} \models F \Rightarrow F \wedge G$	*(absorption)*
$\{F\} \models F \vee G$	*(disjuntion introduction)*

Table 2: Beispiele für Folgerungen (aka inference rules)

Der Begriff der Konsequenz spielt in mehreren Kalkülen eine zentrale Rolle. Eine andere Notation für $\{F_1, \ldots, F_n\} \models G$ ist

$$\begin{array}{l} F_1 \\ F_2 \\ \vdots \\ \underline{F_n} \\ G \end{array} \quad \text{or} \quad \frac{F_1, F_2, \ldots, F_n}{G}$$

Ein Beispiel einen logischen Folgerung ist *modus ponens*:

Lemma 1.31 (*modus ponens*)**.** $\{F, F \Rightarrow G\} \models G$

In den beiden anderen Schreibweisen oben würde der modus ponens so aussehen:

$$\begin{array}{l} F \\ \underline{F \Rightarrow G} \\ G \end{array} \quad \text{, respectively} \quad \frac{F_1, F \Rightarrow G}{G}$$

Im Klartext bedeutet das: Immer wenn F und $F \Rightarrow G$ wahr sind (d. h. wann immer $\mathcal{A} \models F$ und $\mathcal{A} \models F \Rightarrow G$) dann ist auch G wahr — d.h. $\mathcal{A} \models G$). Warum ist das so? Konsequenzregel wahr?

Proof. Wegen Lemma 1.30 müssen wir zeigen, dass $F \wedge (F \Rightarrow G) \Rightarrow G$ eine Tautologie ist:

$$\begin{aligned} F \wedge (F \Rightarrow G) \Rightarrow G &\equiv \neg(F \wedge (\neg F \vee G)) \vee G \equiv \neg F \vee \neg(\neg F \vee G) \vee G \equiv \neg F \vee (F \wedge \neg G) \vee G \\ &\equiv \big((\neg F \vee F) \wedge (\neg F \vee \neg G)\big) \vee G \equiv (\neg F \vee \neg G) \vee G \equiv \neg F \vee \neg G \vee G \end{aligned}$$

Die letzte Formel ist offensichtlich eine Tautologie. Also gilt $F \wedge (F \Rightarrow G) \models G$. □

1.7 Resolutionskalkül

7. Nov.

Ein **Kalkül** ist normalerweise eine Sammlung von Transformationsregeln, oder allgemeiner: jede algorithmische Regel zur Entscheidung einer Frage über logischen Formeln (zum Beispiel "ist $F \equiv G$", oder "ist F erfüllbar?"). Wahrheitswertetafeln sind ein Beispiel für ein Kalkül, aber ein untypisches: Es werden keine Transformationsregeln verwendet. Weitere Beispiele für Kalküle sind solche, die **Inferenzregel**n benutzen. Die Idee ist, aus einer gegebenen Formel (bzw. Menge von Formeln) die logischen Folgerungen zu generieren. Ein bestimmter Kalkül verwendet den Modus Ponens als Inferenzregel. In diesem Abschnitt stellen wir einen weiteren solchen Kalkül vor, der als Inferenzregel die Resolution verwendet.

Später schreiben wir für "G ist mittels eines Kalküls $\mathcal{T}$ aus F gefolgert", bzw. "G ist mittels einer Inferenzregel $\mathcal{T}$ aus F gefolgert" kurz $F \vdash G$. Wenn wir die Inferenzregel betonen wollen, schreiben wir $F \vdash_{\mathcal{T}} G$.

Bemerkung 1.32. Ein Kalkül ist besonders nützlich, wenn er korrekt, vollständig und widerspruchsfrei ist. Die eigentliche Definition benutzt *Axiome*, dazu später mehr. Stellen wir uns hier vor, wir bestimmen, dass die Axiome wahr sein sollen, und dass wir die Axiome (z.B. durch ver-unden) in eine einzige logische Formel F fassen. Dann:

- Der Kalkül ist **korrekt** (engl. *sound*), falls aus $F \vdash G$ immer folgt $F \models G$. (Also für jedes G, das wir mit unserer Regel aus einem wahren F herleiten können, G auch "wahr" ist.)
- Der Kalkül ist **vollständig** (engl. *complete*), falls aus $F \models G$ immer folgt $F \vdash G$. (Also dass jede "wahre" Aussage mit unserer Regel hergeleitet werden kann.)
- Der Kalkül ist **widerspruchsfrei** (engl. *consistent*), falls aus $F \vdash G$ immer folgt $F \nvdash \neg G$. (Da steht also, dass wir mit unserer Regel nicht eine Aussage *und* ihr Gegenteil herleiten können.)

Im Kontext dieses Abschnitts ist unser Ziel, zu entscheiden, ob F (un-)erfüllbar ist. Daher bedeuten die Begriffe hier konkret etwas einfacheres:

- Der Kalkül ist **korrekt**, falls für jedes F, für das er "unerfüllbar" ausgibt, F wirklich unerfüllbar ist.
- Der Kalkül ist **vollständig**, wenn er für jedes unerfüllbare F ausgibt "unerfüllbar".

Das ist ein subtiler Punkt: ein Kalkül, der sowohl korrekt als auch vollständig ist für die Frage "Ist F unerfüllbar?" ist möglicherweise nicht vollständig für die Frage "Ist F erfüllbar?". Z.B. kann es sein, dass er nichts zurückgibt bzw. ewig läuft, falls F erfüllbar ist.

Der Resolutionskalkül verwendet nur eine einzige Transformationsregel, um zu entscheiden, ob eine Formel F erfüllbar oder unerfüllbar ist. (Später werden wir den Resolutionskalkül verwenden, um zu entscheiden, ob eine gegebene Formel in Prädikatenlogik unerfüllbar ist. Dabei tritt der oben geschilderte Effekt auf: falls F unerfüllbar ist, dann wird das immer korrekt beantwortet. Falls F aber erfüllbar ist, kann es passieren, dass der Resolutionskalkül keine Antwort gibt.)

Bemerkung 1.33. Ein Test auf Unerfüllbarkeit von F beantwortet mehrere weitere Fragen: zum Beispiel

- Ist F eine Tautologie? (genau dann, wenn $\neg F$ unerfüllbar ist)
- Ist G eine Folgerung von $F_1, \ldots, F_n$? (genau dann, wenn $F_1 \wedge \cdots \wedge F_n \wedge \neg G$ unerfüllbar ist)

Der erste Punkt ist offensichtlich: $\mathcal{A}(F) = 1$ für alle $\mathcal{A}$ impliziert $\mathcal{A}(\neg F) = 0$ für alle $\mathcal{A}$. Der zweite Punkt ist Lemma 1.30.

Um den Resolutionskalkül für Formeln in Aussagenlogik zu beschreiben brauchen wir etwas Notation. Betrachten wir eine Formel F in KNF:

$$F = (L_{1,1} \vee L_{1,2} \vee \cdots \vee L_{1,n_1}) \wedge (L_{2,1} \vee L_{2,2} \vee \cdots \vee L_{2,n_2}) \wedge \cdots \wedge (L_{k,1} \vee L_{k,2} \vee \cdots \vee L_{k,n_k})$$

wobei die L_{ij} Literale sind. Nun schreiben wir F in **Klauselmengen**-Schreibweise als

$$F = \big\{\{L_{1,1}, L_{1,2}, \ldots, L_{1,n_1}\}, \{L_{2,1}, L_{2,2}, \cdots, L_{2,n_2}\}, \cdots, \{L_{k,1}, L_{k,2}, \cdots, L_{k,n_k}\}\big\}.$$

Verschiedene Formeln können in dieser Schreibweise gleich aussehen. In der Tat beseitigt die Mengenschreibweise Mehrdeutigkeiten. So sind z.B. die Formeln

$$(A_1 \vee \neg A_2) \wedge A_3 \wedge A_3, \quad A_3 \wedge (\neg A_2 \vee A_1), \quad \text{and } A_3 \wedge (\neg A_2 \vee A_1 \vee \neg A_2)$$

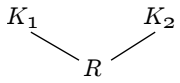
in der Klauselmengenschreibweise alle gleich $\big\{\{A_3\}, \{A_1, \neg A_2\}\big\}$.

Definition 1.34. Seien K_1, K_2 Klauseln, L ein Literal mit $L \in K_1$, $\neg L \in K_2$. Dann ist die **Resolvente** (von K_1 und K_2).

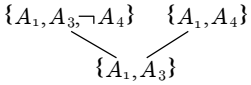
$$R = (K_1 \setminus \{L\}) \cup (K_2 \setminus \{\neg L\})$$

Es kann vorkommen, dass $R = \{\}$ (zum Beispiel, wenn $K_1 = \{A\}$ und $K_2 = \{\neg A\}$). Da diese Situation aber bedeutet, dass die KNF unerfüllbar ist, schreiben wir in diesem Fall $R = \square$ (anstelle von $R = \{\}$ oder $R = \varnothing$, was mit der leeren Klausel verwechselt werden könnte).

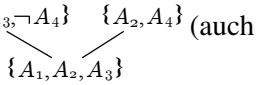
In einem Diagramm schreiben wir die Resolvente R von K_1 und K_2 so:



Beispiel 1.35.



(mit $L = A_4$), oder



(auch mit $L = A_4$).

Es lohnt sich, darüber nachzudenken, ob die gleichzeitige Verwendung zweier unterschiedlicher Literale in einem Resolutionsschritt in Ordnung ist (zum Beispiel A_1 und $\neg A_1$ sowie A_4 und $\neg A_4$).

Lemma 1.36. *Sei F eine Formel, und sei R eine Resolvente. Dann gilt*

$$\{F \vee L, G \vee \neg L\} \models F \vee G$$

(Die Resolution $F \vee G$ ist also eine logische Folgerung von $F \vee L$ und $G \vee \neg L$.) Damit gilt auch: $F \equiv F \cup \{R\}$.

Daher besteht die allgemeine Idee des Resolutionskalküls darin, alle möglichen Resolutionen zu bestimmen. F ist genau dann unerfüllbar wenn $\square$ irgendwann auftaucht.

Notation: Sei F eine Formel in Klauselmengenschreibweise.

- $\mathrm{Res}(F) = F \cup \{R \mid R \text{ Resolvente zweier Klauseln in } F\}$
- $\mathrm{Res}^0(F) = F$
- $\mathrm{Res}^{n+1}(F) = \mathrm{Res}(\mathrm{Res}^n(F))$
- $\mathrm{Res}^*(F) = \bigcup_{n \geq 0} \mathrm{Res}^n(F)$

Auch wenn im Allgemeinen die Fragen "Ist F unerfüllbar?" und "Ist F erfüllbar?" unterschiedlicher Natur sein können, sind sie in der Aussagenlogik bezüglich des Resolutionskalküls gleichweritg:

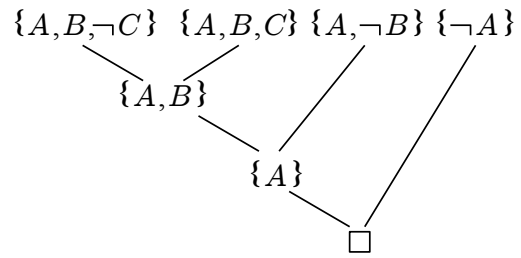
Satz 1.37. *F ist genau dann unerfüllbar, wenn $\square \in \mathrm{Res}^*(F)$. Da die Prozedur immer endet (siehe unten), folgt daraus Der Resolutionskalkül ist korrekt und vollständig.*

Der entsprechende Algorithmus ist jetzt klar: iterativ berechnen wir $\mathrm{Res}^*(F)$. Wir stoppen, wenn $\mathrm{Res}^k(F) = \mathrm{Res}^{k+1}(F)$ für ein k (dann $\mathrm{Res}^k(F) = \mathrm{Res}^*(F)$). Rückgabe "$F$ ist unerfüllbar", wenn $\square \in \mathrm{Res}^*(F)$, return "$F$ ist erfüllbar", sonst.

Der Algorithmus terminiert immer, da wir uns mit ihm befassen nur endliche Formeln. (Im nächsten Abschnitt müssen wir dies anwenden Algorithmus auf unendlich viele Formeln, daher kann es sein, dass er nicht terminiert wenn F erfüllbar ist.) Wir beschreiben den Algorithmus anhand eines Beispiels:

Beispiel 1.38. Gegeben sei $F = (A \vee B \vee \neg C) \wedge (A \vee B \vee C) \wedge (A \vee \neg B) \wedge \neg A$. Wir schreiben F in der Klauselsatznotation:

$$F = \big\{\{A,B,\neg C\},\{A,B,C\},\{A,\neg B\},\{\neg A\}\big\}.$$



Daher ist F unerfüllbar.

Im Allgemeinen handelt es sich bei dem Diagramm möglicherweise nicht um einen Baum: Manchmal muss man die gleiche Klausel mehr als einmal verwenden, wodurch eine Schleife entsteht.

Die im Beispiel verwendeten Resolutionen wurden optimal gewählt. Die Liste aller Resolventen ist:

$$\mathrm{Res}^1(F) = F \cup \big\{\{A,B\},\{A,C\},\{A,\neg C\},\{B,\neg C\},\{B,C\},\{\neg B\}\big\}$$

$$\mathrm{Res}^2(F) = \mathrm{Res}^1(F) \cup \big\{\{A\},\{B\},\{C\},\{\neg C\}\big\}$$

Es wurde viel Arbeit in die Verbesserung des Algorithmus gesteckt, wie Resolutionen auf optimale Weise gewählt werden können. Trotzdem:

Satz 1.39 (Haken 1984)**.** *Es gibt Formeln, bei denen jede Ableitung von $\square$ exponentiell viele Resolutionsschritte benötigt in n, wobei n die Anzahl der atomaren Formeln ist.*

Darüber hinaus kann die KNF einiger gegebener Formel F exponentiell länger sein als F, siehe Bemerkung 1.21.

Bemerkung 1.40. Der Resolutionskalkül ist für Hornformeln effizient: benutze nur diejenigen Resolventen, die die Klauseln K_1 und K_2 verwenden, wobei gilt: K_1 hat nur ein Element, oder K_2 hat nur ein Element. Dies lässt sich durch einen Vergleich der Algorithmen erkennen: Diese Version der Resolution simuliert den Hornformel-Algorithmus 1.3: alle Klauseln mit einem Element decken alle Klauseln der Hornformel von der Form $1 \Rightarrow A_i$ ab. Markieren bedeutet, $\mathcal{A}(A_i) = 1$ zu setzen, also implizit Hinzufügen einer Klausel $\{A_i\}$. Markieren aller Kopien von A_i in $(A_i \wedge A_j \wedge \cdots \wedge A_m \Rightarrow B)$ bedeutet Markierung von A_i in $(\neg A_i \vee \neg A_j \vee \cdots \vee \neg A_m \vee B)$ Dies entspricht der Klausel $\{\neg A_i, \neg A_j, \cdots, \neg A_m, B\}$. Die Resolution der letztgenannten Klausel mit $\{A_i\}$ ist $\{\neg A_j, \cdots, \neg A_m, B\}$ und so weiter.

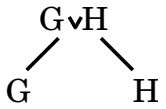
Der Resolutionskalkül ist auch für Formeln (in KNF) effizient, wo jede Klausel höchstens zwei Elemente hat (siehe Übung). Daher ist 2SAT immer effizient entscheidbar

Es gibt viele Implementierungen des Resolutionskalküls und seiner Verfeinerungen (Davis-Putnam-Algorithmus, Davis-Putnam-Logemann-Loveland-Algorithmus, SAT-Löser...)

1.8 Tableaukalkül

Für spätere Zwecke benötigen wir einen weiteren Formalismus zum Entscheiden der Erfüllbarkeit von Formeln in der Aussagenlogik. Es wird davon ausgegangen, dass der Leser vertraut ist mit dem Konzept eines **Binärbaums** vertraut (Wurzel, Scheitelpunkt, Kind, Blatt, Pfad...) Im Folgende bedeutet "Pfad" immer einen Pfad, der in einem Blatt endet.

Die Idee besteht darin, eine Formel F bezüglich ihres rekursiven Aufbaus in ihre Teilformeln aufzudröseln. Die Teilformeln werden nach und nach zu Knoten im Baum. Die Wurzel des Baums ist F. Am Ende sind die Pfade im Baum sind mögliche Belegungen von F. Eine Formel von die Form $G \vee H$ führt zu einer Verzweigung

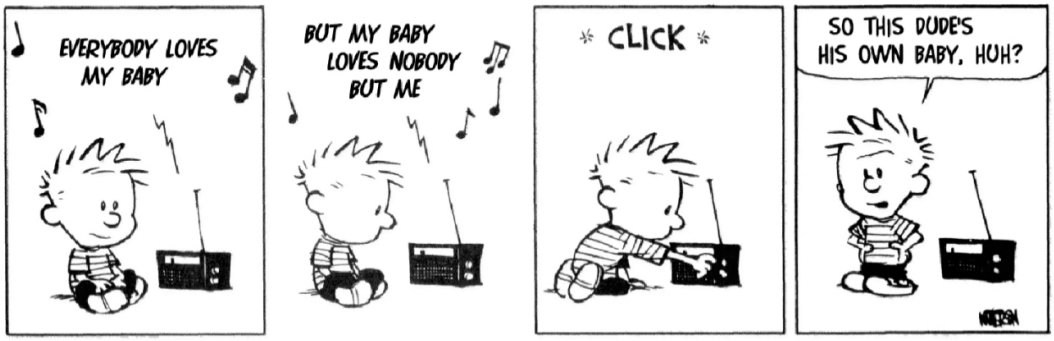


Das bedeutet, dass entweder G oder H (oder beide) wahr sein müssen. Eine Formel von die Form $G \wedge H$ führt zu einer Verzweigung

was bedeutet, dass sowohl F als auch G wahr sein müssen. Das tun wor so lange, bis alle $\wedge$ und $\vee$ in Knoten des Baumes übersetzt sind. Dann sind alle Blätter Literale. Der fertige Baum heißt **Tableau**.

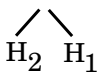
Jetzt überprüfen wir die Erfüllbarkeit: Wenn wir in einem Pfad A_i und $\neg A_i$ sehen, dann ergibt dieser Pfad keine erfüllende Belegung (da in diesem Pfad sowohl A als auch $\neg A$ wahr sein müssen). Eine Formel ist erfüllbar, wenn es einen Pfad gibt, der in diesem Sinne erfüllbar ist. Die allgemeine Methode ist wie folgt.

* * *

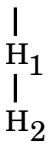


Algorithmus 1.41. Eingabe: eine aussagenlogische Formel F.

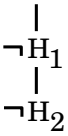
1. Beginne mit F als Wurzel.
2. Wähle einen unmarkierten Knoten G, der kein Literal ist. Markiere G. Wende die folgenden Regeln wiederholt an.
   - Ist G von der Form $\neg\neg H$, dann hänge an jeden Pfad durch G einen Knoten H an.
   - Ist G von der Form $H_1 \vee H_2$, dann hänge an jeden Pfad durch G dies an:

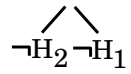
   - Ist G von der Form $H_1 \wedge H_2$, dann hänge an jeden Pfad durch G dies an:

   - Ist G von der Form $\neg(H_1 \vee H_2)$, dann hänge an jeden Pfad durch G dies an:

   - Ist G von der Form $\neg(H_1 \wedge H_2)$, dann hänge an jeden Pfad durch G dies an:

3. Falls es nichts mehr zu mekrieren gibt, gib den Baum (also das Tableau) zurück, STOP.

Ein Pfad ist **geschlossen**, wenn er Knoten A_i und $\neg A_i$ für ein i enthält. In diesem Fall markieren wir das Blatt dieses Pfades mit $\otimes$. Ein Tableau ist **geschlossen**, wenn alle Blätter mit $\otimes$ markiert sind. In diesem Fall ist F unerfüllbar. Sonst ist F erfüllbar, und jeder Pfad mit einem Blatt ohne $\otimes$ liefert für jedes Vorkommen eine erfüllende Belegung $\mathcal{A}$: für jedes Vorkommen von A_i setze $\mathcal{A}(A_i) = 1$, für jedes Vorkommen von $\neg A_i$ setze $\mathcal{A}(A_i) = 0$.

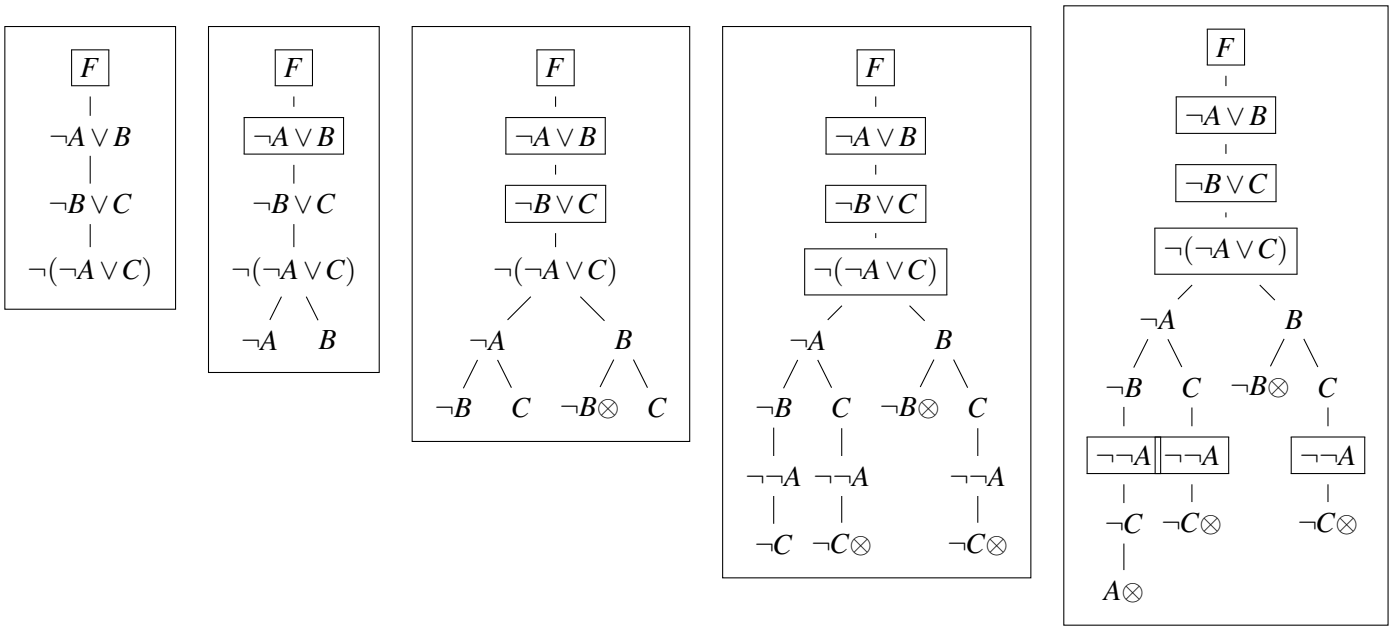


Figure 1: Ein Beispiel für den Tableaualgorithmus. Für jede neue Markierung (hier dargestellt durch einen Kasten um die Formel) gibt es ein neues Bild ((1)-(5)). (Dank an Andreas Mazur)

Es gibt einige offensichtliche Vereinfachungen des Algorithmus. Zum Beispiel kann eine Formel der Form $A \vee B \vee C$ in einem Schritt aufgelöst werden, es werden dann einfach drei Kinder (A, B und C) an jeden Pfad durch diesen Knoten angehängt statt zwei (der Baum ist in diesem Fall kein Binärbaum mehr, aber das stört uns doch nicht, oder?).

Es gibt auch mehrere Verfeinerungen dieses Algorithmus zur Verbesserung die Laufzeit. Zwei offensichtliche Beispiele sind: Pfade können direkt als geschlossen markiert werden, sobald erkannt wird, dass sie geschlossen sind (nicht erst am Ende). Und: Eine Teilformel $H_1 \wedge H_2 \wedge H_3$ kann in einem Schritt abgearbeitet werden, was zu drei neuen Knoten untereinander führt (siehe Beispiel 1.43).

Satz 1.42. *Der Tableau-Algorithmus der Aussagenlogik ist korrekt und vollständig.*

Beispiel 1.43. Wir zeigen, dass $A \Rightarrow C$ eine Folgerung von $\{A \Rightarrow B, B \Rightarrow C\}$ ist. Also $\{A \Rightarrow B, B \Rightarrow C\} \models A \Rightarrow C$. Wegen Bemerkung 1.33 können wir das tun, indem wir zeigen, dass

$$F = (A \Rightarrow B) \wedge (B \Rightarrow C) \wedge \neg(A \Rightarrow C) \equiv (\neg A \vee B) \wedge (\neg B \vee C) \wedge \neg(\neg A \vee C)$$

unerfüllbar ist. Das resultierende Tableau ist in Abbildung 1 dargestellt. Alle Pfade sind geschlossen, daher ist F unerfüllbar, daher folgt die Behauptung.

Zwischenspiel: Relationen

14. Nov

Im nächsten Abschnitt werden Prädikate eine zentrale Rolle spielen. Prädikate sind eine Verallgemeinerung von Relationen. (Prädikate können eins, zwei, drei... Eingaben, wohingegen Relationen immer zwei Eingaben haben.)

Eine Relation wird für eine Menge W erklärt. Stellen Sie sich W z.B. vor als Studierende der Technischen Fakultät, oder alle ganzen Zahlen, oder alle 0-1-Wörter. Im Klartext ist eine Relation eine

Eigenschaft, die ein Paar von Elementen in W kann haben oder nicht. Eine Relation auf der Menge W aller Techfakstudis kann etwa sein "a kennt 'b". Eine Relation auf $W = \mathbb{Z}$ kann zum Beispiel $<$ sein: $2 < 5$ ist wahr, also gilt die Relation für das Paar $(2,5)$. Die Relation gilt nicht für das Paar $(5,2)$ oder das Paar $(3,3)$.

Eine andere Relation auf $W = \mathbb{Z}$ ist z.B. "$a-b$ ist ungerade". Die Relation ist also wahr für $(2,5)$ und $(5,2)$, aber nicht für $(3,3)$

Erinnern Sie sich, dass für zwei Mengen V, W das **kartesische Produkt** dies ist:

$$V \times W := \{(a,b) \mid a \in V, b \in W\}.$$

Eine bequeme Möglichkeit, eine Relation zu definieren, besteht darin, sie als Teilmenge R von $W \times W$ darzustellen. Für das $<$-Beispiel erhalten wir

$$R = \{(a,b) \mid a,b \in \mathbb{Z}, a < b\}$$

Also $(2,5) \in R$, $(5,2) \notin R$, $(3,3) \notin R$. Für das Beispiel "$a-b$ ist ungerade" erhalten wir

$$R' = \{(a,b) \mid (a-b) \bmod 2 = 1\}$$

Also $(2,5) \in R$, $(5,2) \in R$, $(3,3) \notin R$. Eine besonders schöne Klasse von Relationen sind Äquivalenzrelationen.

Definition 1.44. Sei R eine Relation. $R \subseteq W \times W$ heißt **Äquivalenzrelation**, wenn für alle $a,b,c \in W$ gilt:

1. $(a,a) \in R$, *(reflexiv)*
2. $(a,b) \in R$ genau dann, wenn $(b,a) \in R$ und *(symmetric)*
3. $(a,b) \in R$ und $(b,c) \in R$ impliziert $(a,c) \in R$. *(transitiv)*

Eine Äquivalenzrelation unterteilt die Menge W in **Äquivalenzklassen**: Mit $[a]$ bezeichnen wir die Menge aller $b \in W$ mit $(a,b) \in R$. Ein einfaches Beispiel für eine Äquivalenzrelation ist $=$ in $\mathbb{Z}$. Hier besteht jede Äquivalenzklasse $[a]$ nur aus einem Element, nämlich a. Die Beispiele R, R' oben sind beides keine Äquivalenzrelationen. (Warum nicht?) Somit unterteilen sie die Menge $\mathbb{Z}$ auch nicht in sauber getrennte Teilmengen.

2 Prädikatenlogik

In der Aussagenlogik können wir die Aussage "x ist ein Chinese" nicht weiter in Teilaussagen aufteilen. Ebenso wenig die Aussage "für alle $n \in \mathbb{N}$ gilt: $n \geq 0$". Daher erweitern wir die Sprache der Aussagenlogik jetzt um weitere Bausteine wie etwa **Variablen** (wie x). Eine Aussage "x ist ein Chinese" wird dann durch ein **Prädikat** beschrieben, dessen Wert 0 oder 1 ist, je nachdem, ob x Chinese ist oder nicht. Außerdem werden **Quantoren** $\forall$ und $\exists$ hinzugefügt sowie **Funktionen**. Damit erhalten wir die Prädikatenlogik (engl. *first order logic*). Dies ermöglicht die formallogische Behandlung von Aussagen wie etwa

$$\forall \varepsilon \, \exists \delta \, \forall x \, (|x-a| < \delta \Rightarrow |f(x)-f(a)| < \varepsilon)$$

Dabei sind $\forall$ und $\exists$ Quantoren, f und $|\cdot|$ und $-$ sind Funktionen, ε, δ, x und a sind Variablen und $<$ ist ein Prädikat.

Wie in Kapitel 1 ist die allgemeine Struktur dieses Kapitels (1.) formale Definition der Syntax und (2.) der Semantik, (3.) Normalformen, (4.) Algorithmen zur Entscheidung über die (Un-)Erfüllbarkeit von Formeln. Anders als in Kapitel 1 werden wir hier einige Probleme sehen, die algorithmisch unentscheidbar sind (also nicht nur nicht effizient, sondern gar nicht!)

2.1 Syntax der Prädikatenlogik

Wie in Kapitel 1 erklären wir zunächst, was eine korrekte Formel ist und was nicht (Syntax). Danach sehen wir, wie man die Formeln mit konkreter Bedeutung versieht (Semantik).

Definition 2.1 (Syntax der Prädikatenlogik)**.** Die Bausteine von Formeln sind

- **Variablen** sind bezeichnet durch $x_1, x_2, \ldots$ oder u, v, w, x, y, z.
- **Prädikat**-Symbole sind bezeichnet durch $P_1, P_2, \ldots$ oder P, Q, R.
- **Funktions**-Symbole sind bezeichnet durch $f_1, f_2, \ldots$ oder f, g, h.
- **Terme** sind bezeichnet durch $t_1, t_2, \ldots$ und werden induktiv definiert:
  - Jede Variable ist ein Term.
  - Wenn f eine Funktion ist (mit k Eingaben) und $t_1, \ldots, t_k$ sind Terme, dann ist $f(t_1, \ldots, t_k)$ ein Term.
- **Formeln** werden induktiv definiert:
  - Wenn P ein Prädikat ist (mit k Eingaben) und $t_1, \ldots, t_k$ Terme, dann ist $P(t_1, \ldots, t_k)$ eine Formel. (Manchmal nennt man diese *atomare* Formeln)
  - Wenn F eine Formel ist, dann auch $\neg F$.
  - Wenn F und G Formeln sind, dann sind es auch $F \vee G$ und $F \wedge G$.
  - Wenn x eine Variable und F eine Formel ist, dann ist $\forall x F$ und $\exists x F$ sind Formeln.

Alle Zwischenschritte dieser induktiven Konstruktion heißen **Teilformeln**. Ein Auftreten einer Variablen x in einer Teilformel G einer Formel F ist **gebunden**, wenn G die Form hat $\forall x H$ oder $\exists x H$, so dass H x enthält; ansonsten heißt dieses Auftreten von x **frei**. Das heißt, dass eine Variable in

derselben Formel sowohl gebunden als auch frei sein kann, siehe das Beispiel unten. Eine Formel heißt **geschlossen**, wenn alle Variablen nur gebunden auftauchen. Die **Matrix** von F ist das, was man durch Löschen aller $\forall x$ und $\exists x$ erhält. Funktionen ohne Eingaben sind zulässig. Solche Funktionen heißen **Konstanten**.

Bemerkung 2.2. Weil das Konzept "Gleichheit" so natürlich und häufig vorkommt, ist es manchmal sinnvoll, ein weiteres Symbol zuzulassen: $=$. Die Definition der Syntax muss entsprechend angepasst werden: Wenn t_1 und t_2 Terme sind, dann ist $t_1 = t_2$ eine Formel.

Beispiel 2.3. $F = \exists x P(x, f_1(y)) \vee \neg\forall y Q(y, f_7(f_2, f_3(z)))$ ist eine Formel. Alle Teilformeln von F sind

$$F,\ \exists x P(x, f_1(y)),\ P(x, f_1(y)),\ \neg\forall y Q(y, f_7(f_2, f_3(z))),\ \forall y Q(y, f_7(f_2, f_3(z))),\ Q(y, f_7(f_2, f_3(z))).$$

Alle Terme in F sind $x, y, f_1(y), f_7((f_2, f_3(z)), f_2, f_3(z), z$. Alle Auftreten von x in F sind gebunden. Das erste Auftreten von y in F ist frei, das zweite ist gebunden. Das Auftreten von z in F ist frei. Die Matrix von F ist

$$P(x, f_1(y)) \vee \neg Q(y, f_7(f_2, f_3(z))).$$

Bemerkung 2.4. Um weniger Klammern zu verwenden, vereinbaren wir die Regel, dass $\forall x$ und $\exists x$ bindet stärker als $\vee$ und $\wedge$. Daher heißt $\forall x\, P(x) \vee \forall y\, P(y)$ das gleiche wie $(\forall x\, P(x)) \vee (\forall y\, P(y))$, und nicht etwa $\forall x\, (P(x) \vee \forall y\, P(y))$.

2.2 Semantik der Prädikatenlogik

21. Nov.

Um Formeln in der Prädikatenlogik als "wahr" oder "falsch" zu interpretieren, muss man den Symbolen eine Bedeutung geben. Etwas präziser: man muss eine Grundmenge definieren, in der die Variablen und Funktionen ihre Werte annehmen (das Universum), sowie eine Interpretation jedes Prädikatsymbols als Prädikat (z.B. eine Relation) über diesem Universum, eines jeden Funktionsymbols als Funktion usw. Etwas präziser:

Definition 2.5. Eine **Struktur** (auch bekannt als Welt) für eine Formel F ist ein Paar $\mathcal{A} = (U_\mathcal{A}, I_\mathcal{A})$, wobei gilt:

- $U_\mathcal{A}$ ist eine nichtleere Menge (das **Universum**) und
- $I_\mathcal{A}$ ist eine Abbildung (die **Interpretation**), so dass
  - $I_\mathcal{A}(x)$ ein Element von $U_\mathcal{A}$ ist für alle freien Variablensymbole x in F,
  - $I_\mathcal{A}(f)$ eine Funktion über $U_\mathcal{A}$ ist für alle Funktionssymbole f in F,
  - $I_\mathcal{A}(P)$ ein Prädikat über $U_\mathcal{A}$ ist für alle Prädikatsymbole P in F.

Hier hat "Element" die übliche Bedeutung: ein Element von $U_\mathcal{A}$. "Funktion" bezeichnet eine Funktion von $(U_\mathcal{A})^k \to U_\mathcal{A}$, wobei f dann k Eingaben hat. Unter "Prädikat" ist eine Eigenschaft zu verstehen. Z.B., für ein Prädikat P mit einer Eingabe kann $P(x)$ bedeuten: "x ist ungerade", oder "x ist eine Primzahl" (wenn $U_\mathcal{A} = \mathbb{N}$), oder $P(x)$ kann bedeuten "x ist ein Chinese", wenn $U_\mathcal{A}$ die Menge aller Menschen ist. Für ein Prädikat P mit zwei Eingaben kann $P(x, y)$ etwa $x < y$ bedeuten, oder x teilt y (wenn $U_\mathcal{A} = \mathbb{N}$), oder $P(x, y)$ kann bedeuten "x kennt y", wenn $U_\mathcal{A}$ die Menge aller Menschen ist.

Bemerkung 2.6. . Formaler wird ein Prädikat P als Teilmenge von $U_{\mathcal{A}}$ definiert (wenn P eine Eingabe hat) oder eine Teilmenge von $U_{\mathcal{A}} \times U_{\mathcal{A}}$ (wenn P zwei Eingaben hat) oder eine Teilmenge von $U_{\mathcal{A}} \times U_{\mathcal{A}} \times \cdots \times U_{\mathcal{A}}$ (n-mal), wenn P genau n Eingaben hat. Dann ist P zum Beispiel

$$\{x \in \mathbb{N} \mid x \text{ ist ungerade } \}, \quad \{x \in \mathbb{N} \mid x \text{ ist eine Primzahl } \},$$

bzw.

$$\{(n,m) \in \mathbb{N} \times \mathbb{N} \mid n < m\}, \quad \{(n,m) \in \mathbb{N} \times \mathbb{N} \mid n \text{ teilt } m\}$$

für die oben genannten Beispiele.

Im Folgenden werden wir der Kürze halber schreiben $x^{\mathcal{A}}$, $f^{\mathcal{A}}$, $P^{\mathcal{A}}$ anstatt $I_{\mathcal{A}}(x)$, $I_{\mathcal{A}}(f)$, $I_{\mathcal{A}}(P)$.

Beispiel 2.7. Betrachten wir die Formel $F' = (\forall x P(x, f(x))) \wedge Q(g(h,x))$. Hier ist x sowohl eine gebundene Variable als auch eine freie Variable; f ist eine Funktion mit einer Eingabe, g ist eine Funktion mit zwei Eingaben und h ist eine Funktion ohne Eingabe; daher ist h eine Konstante. P ist ein Prädikat mit zwei Eingaben, Q ist ein Prädikat mit einer Eingabe.

Um Verwirrung zu vermeiden (wegen x frei versus x gebunden), betrachten wir $F = (\forall x P(x, f(x))) \wedge Q(g(h,z))$. Eine mögliche Struktur für F ist

$$\begin{aligned} U_{\mathcal{A}} &= \{0,1,2,\ldots\} = \mathbb{N}_0 \\ P^{\mathcal{A}} &= \{(x,y) \in \mathbb{N}_0 \times \mathbb{N}_0 \mid x < y\} \\ Q^{\mathcal{A}} &= \{x \in \mathbb{N}_0 \mid x \text{ ist Primzahl } \} \\ f^{\mathcal{A}} &: \mathbb{N}_0 \to \mathbb{N}_0,\ f(x) = x+1 \\ g^{\mathcal{A}} &: \mathbb{N}_0 \times \mathbb{N}_0 \to \mathbb{N}_0,\ g(x,y) = x+y \\ h^{\mathcal{A}} &= 2 \quad \text{und} \quad z^{\mathcal{A}} = 3. \end{aligned}$$

Dieses $\mathcal{A}$ ist nicht nur eine Struktur für F (also: sie erklärt alles, was in F auftaucht), sondern macht F sogar wahr! Tatsächlich bedeutet F mit dieser Interpretation

$$(\forall x \in \mathbb{N}\, x < x+1) \wedge 2+3 \text{ ist eine Primzahl.}$$

Da beide Aussagen wahr sind, ist F in der Struktur $\mathcal{A}$ wahr. Wir werden in dem Fall kurz $\mathcal{A} \models F$ schreiben.

Wenn wir die obige Struktur in $\mathcal{A}'$ ändern, indem wir $z^{\mathcal{A}} = 3$ ändern zu $z^{\mathcal{A}'} = 4$, dann ist F in der neuen Struktur $\mathcal{A}'$ falsch (da $2+4=6$ keine Primzahl ist).

In der Folge konzentrieren wir uns auf die Frage "Gegeben eine Formel F, gibt es eine Struktur, die F wahr macht?" (Erfüllbarkeit); bzw. "Gegeben eine Formel F, ist F für alle $\mathcal{A}$ wahr?", für $\mathcal{A}$ passend zu F, vergleiche die Definition 1.3 der Tautologie in Kapitel 1. Dies Analogon der 'Tautologie' wird hier aber als "gültig" bezeichnet (oder 'allgemeingültig', aber wir nutzen hier lieber die kurze Form). Was heißt es nun genau, dass $\mathcal{A}$ die Formel F wahr macht:

Definition 2.8. Sei F eine Formel und $\mathcal{A}$ eine Struktur für F. $\mathcal{A}(F)$ wird induktiv definiert durch Definition 1.3 zusammen mit den folgenden Punkten

1. Ist $F = P(t_1, \ldots, t_k)$, dann ist

$$\mathcal{A}(F) = \begin{cases} 1 & \text{falls } (t_1^{\mathcal{A}}, \ldots, t_k^{\mathcal{A}}) \in P^{\mathcal{A}}, \\ 0 & \text{sonst} \end{cases}$$

2. Ist $F = \forall x G$ dann ist

$$\mathcal{A}(F) = \begin{cases} 1 & \text{falls für alle } x^{\mathcal{A}} \in U_{\mathcal{A}} \text{ gilt } \mathcal{A}(G) = 1 \\ 0 & \text{sonst} \end{cases}$$

3. Falls $F = \exists x G$ dann ist

$$\mathcal{A}(F) = \begin{cases} 1 & \text{if there is } x^{\mathcal{A}} \in U_{\mathcal{A}} \text{ with } \mathcal{A}(G) = 1 \\ 0 & \text{sonst} \end{cases}$$

Gegeben sei eine Formel F und eine Struktur $\mathcal{A}$ für F.

- Wenn $\mathcal{A}(F) = 1$, dann **erfüllt** $\mathcal{A}$ die Formel F, kurz: $\mathcal{A} \models F$. Wir sagen in diesem Fall: $\mathcal{A}$ ist ein **Modell** für F.
- Wenn für alle $\mathcal{A}$ für F $\mathcal{A} \models F$ gilt, dann ist F **gültig**. (Dies ist der analoge Begriff der Tautologie in der Aussagenlogik).
- Wenn es eine Struktur $\mathcal{A}$ für F gibt, so dass $\mathcal{A} \models F$, dann ist F **erfüllbar**.

Beispiel 2.9. Sei $F = \forall x \exists y\, P(x,y)$. Eine Struktur $\mathcal{A}$ mit $\mathcal{A} \models F$ ist

$$U_{\mathcal{A}} = \mathbb{N}, \quad P^{\mathcal{A}} = \{(x,y) \mid x < y, x \in \mathbb{N}, y \in \mathbb{N}\}.$$

("Für alle $x \in \mathbb{N}$ existiert $y \in \mathbb{N}$ so dass $x < y$", bingo)

Eine Struktur $\mathcal{A}$ mit $\mathcal{A} \not\models F$ ist etwa

$$U_{\mathcal{A}} = \{0,1,2,\ldots,9\} \quad P^{\mathcal{A}} = \{(x,y) \mid x = 2y, x, y \in U_{\mathcal{A}}\}.$$

In diesem Fall ist F für $x = 1$ falsch.

Um zu verdeutlichen, dass eine Struktur nicht unbedingt eine mathematisches Objekt sein muss: Sei $U_{\mathcal{A}}$ die Menge aller Menschen und $P^{\mathcal{A}}(x,y)$ das Prädikat "x liebt y. Berücksichtigen wir alle Kombinationen der Quantor-Variable Quantor-Variable $P(x,y)$. Dann heißt

- $\forall x\, \exists y\, P(x,y)$ is "jeder liebt jemanden",
- $\forall y\, \exists x\, P(x,y)$ is "jeder wird von jemandem geliebt",
- $\exists x\, \forall y\, P(x,y)$ is "jemand liebt jeden" (Jesus?),
- $\exists y\, \forall x\, P(x,y)$ is "jemand wird von allen geliebt" (Elvis??),
- $\forall x\, \forall y\, P(x,y)$ is "jeder liebt jeden" (Woodstock???),
- $\exists x\, P(x,x)$ is "jemand liebt sich selbst"; usw.

Die Wahrheit jeder Formel hängt von $\mathcal{A}$ ab, aber alle diese Formeln sind erfüllbar (in einem Universum, in dem jeder jeden liebt), und keine der Formeln ist gültig (= wahr in allen möglichen Universen): Wir können einfach $\mathcal{A}$ definieren als eine Struktur, in der niemand niemanden liebt, indem man P als leeres Prädikat wählt.

Bemerkung 2.10. Auch hier gilt für Prädikatenlogik mit Identität, dass die Definition der Semantik angepasst werden muss. Dies geschieht dadurch, dass das Prädikat $=$ hat die übliche Bedeutung hat: gleich sein als Elemente in $U_{\mathcal{A}}$.

Bemerkung 2.11. In Analogie zu Bemerkung 1.33 kann man hier zeigen

- F ist genau dann gültig, wenn $\neg F$ unerfüllbar ist.
- G ist eine Folge von $F_1 \wedge \cdots \wedge F_n$ dann und nur dann, wenn $F_1 \wedge \cdots \wedge F_n \wedge \neg G$ unerfüllbar ist.

Bemerkung 2.12. Die Prädikatenlogik enthält tatsächlich die Aussagenlogik als Sonderfall. Tatsächlich kann jede Formel in Prädikatenlogik ohne Quantoren als Formel in der Aussagenlogik interpretiert werden: Alle Variablen sind frei und werden daher als Konstanten behandelt. Alle Prädikate enthalten nur Konstanten. Daher verschwinden Variablen als solche. Zum Beispiel,

$$F = (P(g) \vee \neg Q(f(g), h)) \wedge R(a, b)$$

entspricht in jeder Interpretation Konstanten und Prädikaten von Konstanten. Letztere sind die atomare Formeln, also wird F zu

$$(A \vee \neg B) \wedge C.$$

Letzteres ist genau dann erfüllbar (bzw. eine Tautologie), wenn ersteres gilt ist erfüllbar (bzw. gültig).

Bemerkung 2.13. Nicht jede Aussage kann in der Prädikatenlogik ausgedrückt werden. Bis jetzt müssen wir uns mit einem Universum begnügen: Wir können nicht "für alle Schüler gibt eine ganze Zahl..." ausdrücken (zumindest nicht so einfach). Darüber hinaus, können wir keine Aussagen wie "für alle Funktionen gilt...", oder "es existiert ein Prädikat, so dass" ausdrücken. Wenn man diese Möglichkeiten berücksichtigt, erhält man *Prädikatenlogik zweiter Stufe*. Letzteres geht über den Rahmen dieses Kurses hinaus.

2.3 Normalformen

Alle Rechengesetze der Aussagenlogik (siehe Satz 1.9) gelten auch für die Prädikatenlogik. Wir brauchen einige weitere Gesetze zur Behandlung von Quantoren. "F ist äquivalent zu G" Die Definition ist hier die gleiche wie in Abschnitt 1: $F \equiv G$ bedeutet, dass für alle $\mathcal{A}$ gilt: $\mathcal{A}(F) = \mathcal{A}(G)$. Gegenüber Kapitel 1 hat sich hier nur die Bedeutung von $\mathcal{A}$ geändert.

Satz 2.14. *Seien F und G Formeln. Dann gelten die folgenden Regeln.*

1. $\neg \forall x\, F \equiv \exists x\, \neg F$
   $\neg \exists x\, F \equiv \forall x\, \neg F$
2. *Falls x nicht frei in G auftaucht, dann*
   $(\forall x\, F) \wedge G \equiv \forall x\, (F \wedge G)$
   $(\forall x\, F) \vee G \equiv \forall x\, (F \vee G)$
   $(\exists x\, F) \wedge G \equiv \exists x\, (F \wedge G)$
   $(\exists x\, F) \vee G \equiv \exists x\, (F \vee G)$
3. $(\forall x\, F) \wedge (\forall x\, G) \equiv \forall x\, (F \wedge G)$
   $(\exists x\, F) \vee (\exists x\, G) \equiv \exists x\, (F \vee G)$
4. $\forall x\, \forall y\, F \equiv \forall y\, \forall x F$
   $\exists x\, \exists y\, F \equiv \exists y\, \exists x F$

Für Prädikatenlogik mit = gelten drei weitere Regeln.

Proof. Die Beweise sind Standard, technisch und liefern keine tieferen Erkenntnisse. Wir zeigen also nur den Beweis der ersten Formel in Punkt 2 des Satzes. Mit Definition 2.8 erhalten wir: 28. Nov

$$\begin{aligned}
\mathcal{A}((\forall x\, F) \wedge G) = 1 \quad & \text{g.d.w. } \mathcal{A}(\forall x\, F) = 1 \text{ und } \mathcal{A}(G) = 1 \\
& \text{g.d.w. } (\text{ für alle } x \in U_{\mathcal{A}} :\ \mathcal{A}(F) = 1) \text{ und } \mathcal{A}(G) = 1 \\
& \text{g.d.w. für alle } x \in U_{\mathcal{A}} :\ (\mathcal{A}(F) = 1 \text{ und } \mathcal{A}(G) = 1) \\
& \qquad (\text{weil } x \text{ nicht frei in } G \text{ vorkommt}) \\
& \text{g.d.w. für alle } x \in U_{\mathcal{A}} :\ \mathcal{A}(F \wedge G) = 1 \\
& \text{g.d.w. } \mathcal{A}(\forall x(F \wedge G)) = 1.
\end{aligned}$$

Dabei heißt "g.d.w." hier: "genau dann, wenn". (Wieder ist der subtile Grund: das Zeichen $\Leftrightarrow$ ist rein auf der syntaktischen Ebene, hier argumentieren wir aber auf einer anderen Ebene.) Alle Regeln oben können in dieser Weise bewiesen werden. □

Analog zu Kapitel benutzen wir die Regeln oben, um eine Normalform einer gegebenen Formel F zu erstellen. Das ist hier allerdings etwas länglich. Als erstes bringen wir die Quantoren alle nach vorn.

Beispiel 2.15.

$$\begin{aligned}
\neg(\exists x\, P(x,y) \vee \forall z\, Q(y)) \wedge \exists w\, Q(w) \equiv & \neg\exists x\, P(x,y) \wedge \neg\forall z\, Q(y) \wedge \exists w\, Q(w) \\
\equiv & \forall x\, \neg P(x,y) \wedge \exists z\, \neg Q(y) \wedge \exists w\, Q(w) \\
\equiv & \forall x\, \big(\neg P(x,y) \wedge \exists z\, \neg Q(y)\big) \wedge \exists w\, Q(w) \\
\equiv & \forall x\, (\exists z(\neg P(x,y) \wedge\ \neg Q(y))) \wedge \exists w\, Q(w) \\
\equiv & \exists w\, \forall x\, \exists z\, (\neg P(x,y) \wedge \neg Q(y) \wedge Q(w))
\end{aligned}$$

Beachten Sie, dass die Reihenfolge der Quantoren im Ergebnis von den gewählten Transformationen abhängt. Insbesondere ist die Reihenfolge nicht immer eindeutig. In diesem Beispiel können die Quantoren sogar in beliebiger Reihenfolge angeordnet werden.

Ein Problem kann auftreten, wenn wir Regel 2 verwenden möchten: $\forall x\, F \wedge G \equiv \forall x\, (F \wedge G)$, aber wenn die Variable x in G frei ist (und gebunden in $\forall x\, F$, also z.B. $\big(\forall x\, P(x)\big) \wedge Q(x)$).

Lemma 2.16. *$F[x/y]$ bezeichne die Formel, die man erhält, wenn man jedes Vorkommen von x in F durch y ersetzt. Sei G eine Formel, die kein y enthält. Dann*

$$\forall x\, G \equiv \forall y\, G[x/y] \quad \textit{und} \quad \exists x\, G \equiv \exists y\, G[x/y]$$

Mit diesem Lemma können wir jede Formel in eine Formel umwandeln, in der keine zwei Quantoren die gleiche Variable haben, und in der keine Variable sowohl gebunden als auch frei vorkommt. Dies ist die Voraussetzung für das Erstellen der folgenden Normalform.

Die nächste Definition beschreibt den oben angedeuteten Schritt im Weg zur endgültigen Normalform. Also, was wir tun wollen, ist: alle Quantoren so weit wie möglich nach links bringen. Machen wir das präzise:

Definition 2.17. Eine Formel F hat **Pränex-Normalform** (PNF), wenn

$$F = Q_1 x_1 Q_2 x_2 \cdots Q_n x_n G,$$

wobei die Q_i Quantoren sind und G keine weiteren Quantoren enthält.

Mit anderen Worten, alle Variablen in G (!) sind frei, bzw. G ist die Matrix von F. Zum Beispiel

$$F = \exists x \, \forall y \, \exists z \, ((P(f(x)) \wedge P(y)) \vee Q(z))$$

hat PNF, wohingegen

$$F = \exists x \, \exists y \, (P(f(x)) \wedge P(y)) \vee \forall z \, Q(z)$$

keine PNF hat.

Satz 2.18. *Jede Formel in der Prädikatenlogik hat eine äquivalente Formel in PNF.*

Auch dieser Satz wird mittels der induktiven Definition einer Formel bewiesen. Lemma 2.16 liefert dabei genug Freiheit, die Variablen so umbenennen zu können, um das zu erreichen.

Für die algorithmische Behandlung von (Erfüllbarkeit von) Formeln benötigen wir aber eine noch speziellere Normalform. Vielleicht haben die Übungen schon eine Idee gegeben, dass es keinen großen Unterschied macht, ob eine Variable frei ist, oder ist durch ein $\exists$ gebunden. Der nächste Schritt besteht daher darin, alle $\exists$ zu entfernen (auf eine angemessene Art und Weise).

Definition 2.19. Gegeben sei eine Formel in PNF. Die **Skolem-Normalform** (SNF) davon ist die Ausgabe des folgenden Algorithmus:

1. während F $\exists$ enthält, tue:
   - Ist $F = \forall y_1 \, \forall y_2 \, \cdots \forall y_n \, \exists x \, G$
     Dann wähle eine Funktion f, die nicht in G enthalten ist, und setze

     $$F := \forall y_1 \, \forall y_2 \, \cdots \forall y_n \;\; G[x/f(y_1, \ldots, y_n)].$$

2. STOP

Beachten Sie, dass G ein $\forall$ enthalten kann. Zum Beispiel, wenn $F = \forall x \exists y \forall z \, P(x,y,z)$, dann ist $G = \forall z \, P(x,y,z)$, und F wird zu $\forall x \forall z \, P(x, f(x), z)$.

Der Fall $n = 0$ ist möglich, das heißt, F hat die Form $\exists x G$. In diesem Fall hat f keine Argumente, daher ist f eine Konstante. Also wird in diesem Fall F durch $G[x/f]$ ersetzt. Einen ähnlichen Fall sehen wir in Beispiel 2.23 unten.

Satz 2.20. *Für jede Formel F in der Prädikatenlogik gibt es eine Formel G in SNF, so dass F genau dann erfüllbar ist, wenn G es ist.*

Definition 2.19 liefert uns ja das Rezept, wie wir das G bekommen. Nach diesem Schritt bringen wir die Matrix der Formel noch auf KNF (siehe Kapitel 1, Def. 1.12).

Bemerkung 2.21. Offenbar spielen Konstanten hier eine besondere Rolle. Bis jetzt haben wir Funktionssymbole $f, g, h...$ für Konstanten benutzt (wobei $f, g, h...$ Funktionen mit null Eingaben sind). Der besseren Lesbarkeit halber verwenden wir ab jetzt $a, b, c...$ oder $a_1, a_2, ...$ für Konstanten.

Die folgende Liste fasst alle Schritte zusammen, die zum Transformieren einer beliebigen Formel in die Normalform, die wir später brauchen werden, erforderlich sind. Für diese Normalform scheint es in der Literatur keinen bestimmten Namen zu geben, obwohl sie im Folgenden total wichtig wird. Deshalb nennen wir sie die normale Normalform.

Algorithmus 2.22 (**Die normale Normalform (NNF)**). Gegeben eine Formel.

0. Ersetze alle Teilformeln $F \Rightarrow G$ by $\neg F \vee G$, und alle $F \Leftrightarrow G$ durch $(F \wedge G) \vee (\neg F \wedge \neg G)$.
1. Benenne alle gebundenen Variablen in F um, bis keine zwei Quantoren dieselbe Variable haben, und bis keine Variable sowohl gebunden als auch frei erscheint.
2. Seien $y_1, \ldots, y_n$ alle freien Variablen in F. Ersetze jedes y_i durch eine Konstante a_i.
3. Forme in PNF um (Quantoren nach vorn).
4. Enferne alle $\exists$ durch Umformen in SNF.
5. Forme die Matrix der Formel in KNF um.

Ausgabe: die so entstandene Formel (mit den Allquantoren, also nicht nur die Matrix)

Beachten Sie, dass nur die Schritte 0, 1, 3 und 5 die Äquivalenz von Formeln erhalten. Im Allgemeinen wird die Ausgabe am Ende keine Formel sein, die zur ursprünglichen Formel äquivalent ist. Das Ergebnis ist aber genau dann erfüllbar, falls die ursprüngliche Formel es ist.

Beispiel 2.23. Ein vollständiges Beispiel, das alle Schritte zur Erstellung der NNF veranschaulicht. Sei

$$F = \exists x \forall y \left(P(x) \Rightarrow Q(y)\right) \wedge P(x) \wedge \forall y \neg Q(y).$$

Anwenden von Schritten 0-5 liefert:

$$\begin{aligned}
F &\overset{0.}{\equiv} \exists x \forall y \left(\neg P(x) \vee Q(y)\right) \wedge P(x) \wedge \forall y \neg Q(y) \\
&\overset{1.}{\equiv} \exists w \forall y \left(\neg P(w) \vee Q(y)\right) \wedge P(x) \wedge \forall z \neg Q(z) \\
&\overset{2.}{\cong} \exists w \forall y \left(\neg P(w) \vee Q(y)\right) \wedge P(a) \wedge \forall z \neg Q(z) \\
&\overset{3.}{\equiv} \forall z \exists w \forall y \left(\left(\neg P(w) \vee Q(y)\right) \wedge P(a) \wedge \neg Q(z)\right) \\
&\overset{4.}{\cong} \forall z \forall y \left(\left(\neg P(f(z)) \vee Q(y)\right) \wedge P(a) \wedge \neg Q(z)\right)
\end{aligned}$$

Die Matrix der letzten Formel hat bereits KNF, also müssen wir für Schritt 5 nichts tun. Das Zeichen $F \cong G$ benutzen wir hier provisorisch für "F erfüllbar genau dann wenn G erfüllbar".

2.4 Resolutionskalkül der Prädikatenlogik

5. Dez

Wir werden im Abschnitt 4 sehen, dass wir auf nichts Besseres hoffen können als auf einen Test auf Unerfüllbarkeit einer Formel F. Ein solcher Test gibt "Ja" zurück, wenn F unerfüllbar ist, und gibt möglicherweise nichts zurück (läuft ewig) wenn F erfüllbar ist. Darüber hinaus kann (wird!) es passieren, dass wir auf eine positive Antwort beliebig lang warten: wenn die Wartezeit beschränkt wäre (irgendwie, z.B. exponentiell in der Länge der Formel, oder überexponentiell), dann könnte ja dieser Test in einen Test auf Erfüllbarkeit umgewandelt werden ("lange genug warten, und wenn der Algorithmus nicht terminiert, gib 'erfüllbar' zurück"). Der Test wäre halt nur nicht effizient.

Das Problem besteht im Wesentlichen darin, dass wir unendlich viele Strukturen testen müssen, unter anderem aufgrund des folgenden Ergebnisses:

Satz 2.24. *Es gibt erfüllbare Formeln in der Prädikatenlogik, die auschließlich unendliche Modelle haben. Das heißt, Modelle $\mathcal{A} = (U_\mathcal{A}, I_\mathcal{A})$ so dass $U_\mathcal{A}$ eine unendliche Menge ist.*

Proof. (Übung) □

Die Lösung des Problems besteht darin, eine bestimmte Standardstruktur zu verwenden.

Definition 2.25. Das **Herbrand-Universum** $H(F)$ einer Formel F ist die Menge aller variablenfreien Terme, die aus den Teilformeln von F konstruiert werden kann. Das heißt, $H(F)$ besteht aus

1. allen in F vorkommenden Konstanten $a, b, \ldots$, sowie:
2. wenn $t_1, t_2, \ldots t_n \in H(F)$, und f in F vorkommt (wobei f genau n Eingaben hat), dann ist auch $f(t_1, t_2, \ldots, t_n) \in H(F)$.

Wenn F keine Konstante enthält, dann füge man eine Konstante a zu $H(F)$ hinzu und verfahre wie oben.

Beachten Sie den Dreh hier: Die Elemente des Universums sind die Symbole in F. Das bedeutet, dass wir die Semantik als die Syntax *definieren*. Also sind zwei beliebige unterschiedliche Symbole unterschiedliche Elemente von $H(F)$.

Beispiel 2.26. Sei $F = \forall x\, \forall y\, \forall z\, P(x, f(y), g(z,x))$. F enthält keine Konstante, also fügen wir a zu $H(F)$ hinzu und erhalten

$$H(F) = \{a, f(a), g(a,a), f(f(a)), f(g(a,a)), g(a, f(a)), g(f(a), a), g(f(a), f(a)), f(g(a, f(a))), \ldots\}$$

Sei $G = \forall x\, \forall y\, Q(a, f(z), h(y,b))$. G enthält zwei Konstanten a, b, und wir erhalten

$$\begin{aligned} H(G) = \{&a, b, f(a), f(b), h(a,a), h(a,b), \\ &h(b,b), h(b,a), f(f(a)), f(f(b)), \\ &f(h(a,a)), f(h(a,b)), h(f(a), b), \\ &h(f(b), b), h(a, f(a)), h(f(a), f(a)), ...\} \end{aligned}$$

Beachten Sie im letzten Beispiel, dass auch z.B. $h(a,a)$ zu $H(G)$ gehört, obwohl dass h nur als $h(y,b)$ in G vorkommt. Um eine Struktur $\mathcal{A}$ zu erhalten, benötigen wir noch eine Interpretation. Alle Terme sind bereits definiert (z. B. $a^\mathcal{A} = a, f^\mathcal{A}(a) = f(a)$ und so weiter). Eigentlich müssten wir noch die Interpretation der Prädikate definieren. Aber im Folgenden werden wir die konkreten Interpretationen der Prädikate ignorieren. Wir iterieren über alle Möglichkeiten, wie die Prädikate mit Termen aus $H(F)$ belegt sein können — also etwa

$$P^\mathcal{A}(a, f(a), g(a,a)), P^\mathcal{A}(b, f(a), g(a,a)), P^\mathcal{A}(a, f(b), g(a,a)), P^\mathcal{A}(a, f(a), g(b,a)), \ldots$$

im obigen Beispiel — und fassen alle diese Instanzen als atomare Formeln auf, bis wir einen Widerspruch erhalten.

Jede Struktur $\mathcal{A} = (H(F), I_\mathcal{A})$ mit $\mathcal{A} \models F$ nennen wir ein **Herbrand-Modell** für F.

Satz 2.27. *Sei F in Skolem-Normalform (bzw. in NNF). Dann ist F genau dann erfüllbar, wenn F ein Herbrand-Modell hat.*

Ein Beweis findet sich im Buch von Schöning. Der Beweis ist für die Skolemnormalform ausgeführt, für die NNF ist es dann klar (weil die Matrix der Formel äquivalent ist zu ihrer KNF).

Definition 2.28. Sei $F = \forall y_1 \cdots \forall y_n F^*$ eine Formel im Skolem-Normal Form, und sei F^* die Matrix von F. Sei $E(F)$ die Menge von allen Instanzen von F, das heißt:

$$E(F) = \{F^*[y_1/t_1][y_2/t_2]\cdots[y_n/t_n] \mid t_1, t_2, \ldots, t_n \in H(F)\}.$$

$E(F)$ heißt **Herbrand-Expansion**.

Satz 2.29. *Für jede Formel F in Skolem-Normalform (bzw. NNF) gilt: F ist erfüllbar genau dann, wenn $E(F)$ erfüllbar ist (im Sinne der Aussagenlogik).*

Beachten Sie, dass $E(F)$ unendlich ist. Also ist dies der Punkt, an dem der Kompaktheitssatz (Thm. 1.27) benötigt wird. Jetzt sind wir in der Lage, einen Test bereitzustellen, der beantwortet, ob eine gegebene Formel unerfüllbar ist.

Algorithmus 2.30. Eingabe: eine Formel F in normaler Normalform (NNF). Sei $E(F) = \{F_1, F_2, F_3, \ldots\}$ eine Aufzählung der Herbrand-Expansion von F. Sei $n = 0$, $M := \{\}$.

while $\square \notin M$ do

- $n := n + 1$
- $M := M \cup \{F_n\}$
- $M := \mathrm{Res}^*(M)$.

Return "unerfüllbar".

Man beachte, dass dieser Test ewig läuft, falls F erfüllbar ist. (In dem Sinne ist es also streng genommen kein Algorithmus.)

Da $\neg F$ genau dann unerfüllbar ist, wenn F gültig ist, liefert das auch ein Verfahren zum Testen, ob eine gegebene Formel gültig ist (aber nicht, ob sie ungültig ist, analog zu oben). Genauso bekommen wir damit einen Test für $F \models G$ (also G ist Folgerung von F), siehe Bemerkung 2.11.

Beispiel 2.31. Betrachten wir $F = \forall x\,(P(x) \wedge \neg P(f(x))$. Die Matrix von F in der Klauselmengennotation ist $\{\{P(x)\}, \{\neg P(f(x))\}\}$. Das Herbrand-Universum ist $H(F) = \{a, f(a), f(f(a)), f(f(f(a))), \ldots\}$.
Die Herbrand-Expansion ist $E(F) = \Big\{\{\{P(a)\}, \{\neg P(f(a))\}\}, \{\{P(f(a))\}, \{\neg P(f(f(a)))\}\}, \ldots\Big\}$.
Wir erhalten

$$\begin{array}{cccc} \{P(a)\} & \{\neg P(f(a))\} & \{P(f(a))\} & \{\neg P(f(f(a)))\} \\ & \searrow \quad \swarrow & & \\ & \square & & \end{array}$$

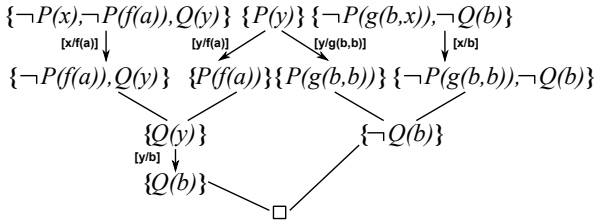
Dies zeigt, dass F unerfüllbar ist.

Im obigen Beispiel benötigten wir nur vier Klauseln und nur einen Resolutionsschritt. Dennoch sind zwei der generierten Klauseln überflüssig. In größeren Beispielen zeigt sich, dass es sich lohnt, nur Klauseln zu erstellen, von denen zu erwarten ist, dass sie schell zu $\square$ führen.

Beispiel 2.32. Betrachten Sie $F = \forall x \forall y \big((\neg P(x) \vee \neg P(f(a)) \vee Q(y)) \wedge P(y) \wedge (\neg P(g(b,x)) \vee \neg Q(b))\big)$.
Die Matrix von F in der Klauselmengennotation ist

$$\{\{(\neg P(x), \neg P(f(a)), Q(y)\}, \{P(y)\}, \{\neg P(g(b,X)), \neg Q(b)\}\}$$

Nun versuchen wir klug zu resolvieren: Wir beginnen mit der Matrix von F und ersetzen die Variablen auf vorausschauende Weise:



Daher ist F unerfüllbar.

Die Richtungen, die man an dieser Stelle weiter verfolgen würde, sind

1. Wie können wir diesen Algorithmus verfeinern, um ihn noch besser zu machen? Z.B. Verwendung vorausschauender oder verzögerter Substitutionen (wie die Verzögerung von $[y/b]$ im obigen Beispiel)
2. Welche Klassen von Formeln sind effizient entscheidbar? (vergleiche Hornformeln)
3. Können wir den Resolutionkalkül anwenden, um mathematische Aussagen automatisch zu beweisen?

Bemerkung 2.33. Um den letzten Punkt zu veranschaulichen, betrachten wir diese Formel:

$$F = \forall x\, \forall y\, \forall z\, f(x, f(y,z)) = f(f(x,y),z) \wedge \forall x\, f(x,e) = x \wedge \forall x\, \exists y\, f(x,y) = e$$

Jedes Modell für F ist eine Gruppe (siehe Mathematik I oder Wikipedia). Daher ist jede aus F abgeleitete Konsequenz eine (wahre) Aussage über Gruppen. Wenn es eine Maschine (oder einen Algorithmus) gibt, die nun nacheinander alle wahren Aussage über Gruppen ausspuckt, werden viele Mathematiker (nämlich alle Gruppentheoretiker) arbeitslos. Daran wird gearbeitet, es ist jedoch unklar ob es gelingen wird. Im Moment lautet die Annahme "Nein". Ein Problem ist, dass eine Maschine nicht erkennen kann, welches Ergebnis "wichtig" ist und welches nicht. Die Tendenz geht in Richtung "computergestützte Beweise" und "computergeprüfte Beweise", siehe z.B. den sehr lesenswerten Übersichtsartikel Formal Proof von Thomas Hales. (Beachten Sie auch die Anzeige am Ende der Arbeit!) 12. Dez

3 Modale Logik

In manchen Kontexten ist es wünschenswert, die Bedeutungen von "wahr" und "falsch" zu verallgemeinern. Bei den Aussagen "ich habe Hunger" oder "ich trage ein blaues Hemd" hängt das ja von den Umständen ab, z.B. der Tages- oder Wochenzeit. Bezüglich der chronologischen Reihenfolge können wir unterscheiden

- A ist immer wahr vs
- A ist manchmal wahr

Oder hinsichtlich der Konsequenz können wir unterscheiden

- A ist notwendigerweise wahr vs
- A ist möglicherweise wahr

Zu diesem Zweck führen wir zwei neue Operatoren ein: $\Box$ und $\diamond$.

3.1 Syntax und Semantik

Die hier diskutierte Modallogik ist eine Erweiterung der Aussagenlogik. Es gibt auch eine Modallogik erster Ordnung, die jedoch nicht Teil dieses Kurses ist.

Definition 3.1 (Syntax)**.** Eine Formel in der Modallogik wird induktiv wie folgt definiert:

1. Jede Formel in der Aussagenlogik ist eine Formel in der Modallogik.
2. Wenn F und G Formeln in der Modallogik sind, sind dann $\neg F$, $F \vee G$, $F \wedge G$, $\Box F$ ("notwendig F") und $\diamond F$ ("möglich F") Formeln in der Modallogik.

Zum Beispiel ist $\diamond A \Rightarrow \neg\Box(B \wedge \diamond\neg C)$ eine Formel (in der Modallogik; im Rest dieses Abschnitts bedeutet "Formel" immer "Formel in Modallogik", wenn nicht ausdrücklich abders angegeben).

Um nicht zu viele Klammern zu verwenden, vereinbaren wir, dass $\diamond$ und $\Box$ stärker binden als $\wedge$ und $\vee$ (daher stärker als $\Rightarrow$ und $\Leftrightarrow$).

Beispiel 3.2. Drei Fußballexperten treffen drei Aussagen:

1. "Wenn Schalke jemals wieder die Meisterschaft gewinnt, werde ich einen Besen fressen."
2. "Irgendwann wird deine Aussage wahr."
3. "Wenn Aussage 2 richtig ist, ist das dasselbe, als würde man sagen, dass Schalke von nun an immer Meister wird."

Wenn A="Schalke wird Meister" und B="Experte 1 wird einen Besen fressen" dann kann 1. als $\diamond A \Rightarrow \diamond B$ angegeben werden, 2. wird $\diamond(\diamond A \Rightarrow \diamond B)$, und 3. wird zu $\diamond(\diamond A \Rightarrow \diamond B) \Leftrightarrow \Box A$. (Es ist noch nicht klar, ob diese Aussagen wahr sind oder überhaupt Sinn ergeben.)

Um die Semantik der Modallogik zu definieren, brauchen wir mehrere Zutaten: eine Menge W verschiedener Zustände ("Welten"), in denen eine Formel jeweils gilt oder nicht gilt. Für jeden Zustand in W gibt es weitere Punkte in W, die von diesem Zustand aus erreicht werden können ("mögliche Zukünfte"), andere nicht (z. B. "die Vergangenheit" oder "unmögliche Zukünfte"). Dies wird wie folgt realisiert.

Definition 3.3 (Semantik)**.** Eine **Struktur** (in der Modallogik) ist ein Tripel $\mathcal{A} = (W, R, \alpha)$, wobei gilt:

- (W, R) ist ein Paar, so dass W eine nichtleere Menge ist und R eine Relation auf W (also ist $R \subset W \times W$). Das Paar (W, R) wird **Rahmen** genannt.
- Sei M eine Menge von atomaren Formeln. Eine Abbildung $\alpha : M \times W \to \{0, 1\}$ ist eine **Belegung** von M. (Im Klartext: α weist jedem Paar (A, s) wahr (1) oder falsch (0) zu.)

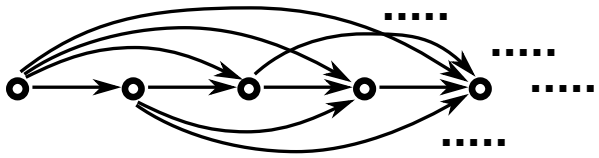
Das Paar (W,R) wird als **Rahmen** für F bezeichnet. (W,R) kann dargestellt als *gerichteter Graph*, siehe unten.

Nun können wir Wahrheitswerte für eine Formel F induktiv definieren.

1. Ist $F = A_i$ eine atomare Formel, dann $\mathcal{A}(F,s) = \alpha(A_i,s)$.
2. Ist F eine Formel dann $\mathcal{A}(\neg F,s) = \begin{cases} 1 & \text{if } \mathcal{A}(F,s) = 0 \\ 0 & \text{sonst} \end{cases}$
3. Sind F und G Formeln, dann $\mathcal{A}(F \wedge G,s) = \begin{cases} 1 & \text{if } \mathcal{A}(F,s) = \mathcal{A}(G,s) = 1 \\ 0 & \text{sonst} \end{cases}$
4. Sind F und G Formeln, dann $\mathcal{A}(F \vee G,s) = \begin{cases} 1 & \text{if } \mathcal{A}(F,s) = 1 \text{ or } \mathcal{A}(G,s) = 1 \\ 0 & \text{sonst} \end{cases}$
5. Ist F eine Formel, dann $\mathcal{A}(\Box F,s) = \begin{cases} 1 & \text{if } \mathcal{A}(F,t) = 1 \text{ for all } t \in W \text{ with } (s,t) \in R \\ 0 & \text{sonst} \end{cases}$
6. Ist F eine Formel, dann $\mathcal{A}(\diamond F,s) = \begin{cases} 1 & \text{if } \mathcal{A}(F,t) = 1 \text{ for some } t \in W \text{ with } (s,t) \in R \\ 0 & \text{sonst} \end{cases}$

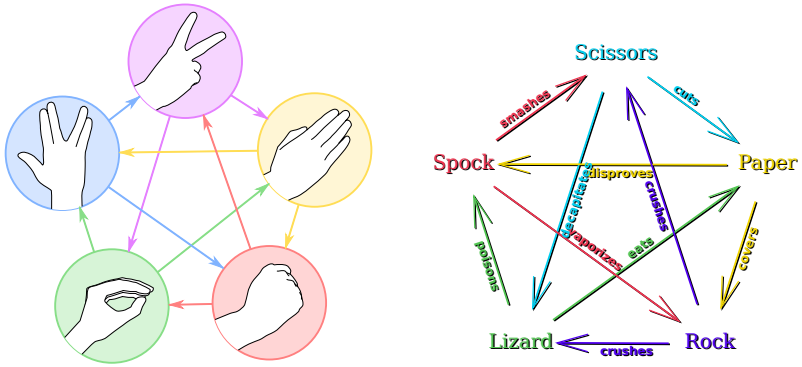
Wie immer gehen wir implizit davon aus, dass $\mathcal{A}$ bzw. M alle atomaren Formeln von F enthält.

Beispiel 3.4. Ein Rahmen (W,R) kann als **gerichteter Graph** visualisiert werden. Die Elemente von W sind die Knoten des Graphen, und ein Element $(n,m) \in R$ ergibt eine gerichtete Kante von n nach m. So sieht der Graph für die Relation im obigen Beispiel, d. h. für $(W,R) = (\mathbb{N}_0,<)$, teilweise wie folgt aus:



Insbesondere ist der obige Graph ein unendlicher Graph. Darüber hinaus gehen von jedem Knoten unendlich viele Kanten weg. Ein einfacheres Beispiel für einen zeitlichen Rahmen wären die Wochentage $\{1,2,\ldots,7\}$, geordnet mittels $<$.

Ein weiteres einfaches Beispiel ist das Spiel Stein-Papier-Schere-Spock-Eidechse. Hier $W = \{$Stein, Papier, Schere, Eidechse, Spock$\}$, und $R = \{(n,m) \mid n,m \in W,\ n \text{ schlägt } m\}$. Die entsprechende Diagramm ist (zwei Versionen)



$\mathcal{A}$ und s gegeben, $\mathcal{A}(F,s)=1$	$s \Vdash_{\mathcal{A}} F$	F *gilt* in s
There is $\mathcal{A}, s$ with $\mathcal{A}(F,s)=1$	—	F ist *erfüllbar*
$\mathcal{A}$ gegeben, für alle $s \in W:\ \mathcal{A}(F,s)=1$	$\mathcal{A} \models F$	$\mathcal{A}$ ist *Modell* für F
(W,R) gegeben, für alle $\alpha, s \in W:\ \mathcal{A}(F,s)=1$	$(W,R) \models F$	F ist *gültig* (in (W,R))
Für alle $\mathcal{A}=(W,R,\alpha), s \in W:\ \mathcal{A}(F,s)=1$	$\models F$	F ist eine *Tautologie*.

Table 3: Ein Überblick darüber, auf welche Weisen eine Formula F in Modallogik "wahr" sein kann, abhängig davon, was genau in $\mathcal{A}=(W,R,\alpha), s \in W$ festgelegt ist (vgl. Def. 3.6).

Beispiel 3.5. (Bsp. 3.2 Fortsetzung) Eine mögliche Struktur für $F = \diamond(\diamond A \Rightarrow \diamond B) \Leftrightarrow \Box A$ ist etwa $\mathcal{A}=(W,R,\alpha)$, wobei $W=\mathbb{N}_0$ (wobei 0 für die aktuelle Meisterschaft 2023/24 steht, 1 für die nächste Meisterschaft usw.), die Relation R ist $<$ (d. h. $R=\{(n,m) \mid n,m \in \mathbb{N}_0, n<m)\}$). Eine Belegung α, die F wahr macht, ist

$$\alpha(A,s)=1 \text{ für alle } s \in \mathbb{N}_0, \quad \alpha(B,s)=1 \text{ für } s \text{ odd}, \quad \alpha(B,s)=0 \text{ sonst.}$$

("Schalke wird von nun an immer Meister werden", "Experte 1 frisst einen Besen in Saison 2024/25, 2026/27, 2028/29...")

Eine andere Struktur für F — mit demselben Rahmen — ist $\mathcal{A}'=(\mathbb{N}_0,<,\beta)$ ist zum Beispiel gegeben durch

$$\beta(A,s)=1 \text{ für } s=2,\ \beta(A,s)=0 \text{ sonst}, \quad \beta(B,s)=1 \text{ für } s=3,\ \beta(B,s)=0 \text{ sonst.}$$

In diesem Fall $\mathcal{A}'(\diamond(\diamond A \Rightarrow \diamond B),s)=\mathcal{A}'(\diamond(\neg\diamond A \vee \diamond B),s)=1$ für alle $s \geq 2$ (da $\mathcal{A}'(\diamond A,s)=0$ für $s \geq 2$, also $\mathcal{A}'(\neg\diamond A,s)=1$ für $s \geq 2$, also $\mathcal{A}'(\diamond(\neg\diamond A \vee \diamond B),t)=1$ für alle t). Andererseits gilt $\mathcal{A}'(\Box A,s)=0$ für alle s. Daher ist $\mathcal{A}'(F,s)=0$ für alle $s \geq 2$ in dieser Struktur.

Insbesondere hängt die Wahrheit einer Formel F von dem Punkt s ab, in dem wir F auswerten. Daher werden wir die Begriffe "Modell" und "gültig" nun anpassen an die Modallogik. Die Option, verschiedene Zeitpunkte, Belegungen, und Rahmen für F auszuwählen, liefert viele verschiedene
19. Dez Bedeutungen für "F ist wahr". Die folgende Definition benennt die vielen Möglichkeiten.

Definition 3.6. Sei F eine Formel in der Modallogik und $\mathcal{A}=(W,R,\alpha)$ eine Struktur für F.

- Wenn $\mathcal{A}(F,s)=1$, dann **gilt** F in s. (Man schreibt kurz $s \Vdash_{\mathcal{A}} F$, oder $s \Vdash F$, wenn $\mathcal{A}$ aus dem Kontext klar ist). Wenn es ein $\mathcal{A}=(W,R,\alpha)$ und ein $s \in W$ gibt mit $s \Vdash_{\mathcal{A}} F$, dann heißt F **erfüllbar**.
- Wenn $\mathcal{A}(F,s)=1$ für alle $s \in W$, nennen wir $\mathcal{A}$ ein **Modell** für F, kurz: $\mathcal{A} \models F$.
- Es sei ein Rahmen (W,R) gegeben. Wenn für alle Strukturen $\mathcal{A}=(W,R,\alpha)$ gilt $\mathcal{A} \models F$, dann heißt F **gültig** in (W,R). In diesem Fall schreiben wir kurz $(W,R) \models F$.
- F ist eine **Tautologie**, wenn F in jedem Rahmen für F gültig ist.

Bezüglich Beispiel 3.5 oben: mit $\mathcal{A}=(\mathbb{N}_0,<,\alpha)$ erhalten wir, dass $\mathcal{A}(F,s)=1$ für alle $s \in \mathbb{N}_0$, also ist $\mathcal{A}$ ein Modell für F, kurz: $\mathcal{A} \models F$.

Mit $\mathcal{A}'=(\mathbb{N}_0,<,\beta)$ haben wir gesehen, dass $\mathcal{A}'(F,s)=0$ für alle $s \geq 2$. Daher $\mathcal{A}' \not\models F$. Deshalb sehen wir auch $(N_0,<) \not\models F$, das heißt, F ist in $(\mathbb{N}_0,<)$ nicht gültig. Deshalb ist F erst recht keine Tautologie.

3.2 Rechenregeln und (keine) Normalformen

Wie in den Abschnitten 1 und 2 haben wir auch einige Rechenregeln für die Modallogik. Um diese anzugeben, müssen wir Äquivalenz und Folgerung definieren.

Definition 3.7. G ist eine **Folgerung** von F (kurz: $F \models G$), wenn für alle $\mathcal{A}$ mit $\mathcal{A} \models F$ gilt: $\mathcal{A} \models G$. F ist **äquivalent** zu G, wenn für alle $\mathcal{A}$ gilt: $\mathcal{A} \models F$ genau dann, wenn $\mathcal{A} \models G$. In diesem Fall schreiben wir kurz $F \equiv G$.

Bemerkung 3.8. Wie in Lemma 1.30 gilt $F \models G$ genau dann, wenn $F \Rightarrow G$ eine Tautologie ist, genau dann, wenn $F \wedge \neg G$ unerfüllbar ist.

Im Folgende konzentrieren wir uns daher wieder hauptsächlich auf die übliche Frage "Ist F unerfüllbar?" (oder nicht!)

Satz 3.9. *Seien F, G zwei Formeln. Zusätzlich zu allen Berechnungsregeln für Formeln in der Aussagenlogik (siehe Satz 1.9) gelten folgende Regeln:*

1. $\neg\Box F \equiv \Diamond\neg F$
2. $\neg\Diamond F \equiv \Box\neg F$
3. $\Box(F \Rightarrow G) \models \Box F \Rightarrow \Box G$
4. $\Box(F \Rightarrow G) \models \Diamond F \Rightarrow \Diamond G$
5. $\Diamond(F \Rightarrow G) \equiv \Box F \Rightarrow \Diamond G$
6. $\Box(F \wedge G) \equiv \Box F \wedge \Box G$
7. $\Diamond(F \vee G) \equiv \Diamond F \vee \Diamond G$
8. *Wenn F gültig ist, dann ist $\Box F$ gültig.*

Alle Aussagen können wieder durch die induktive Definition von Wahrheitswerten bewiesen werden. Regel 1 kann beispielsweise wie folgt gezeigt werden:

$$\begin{aligned}
\mathcal{A}(\neg\Box F, t) = 1 &\equiv \neg(\mathcal{A}(\Box F, t) = 1) \\
&\equiv \text{ nicht für alle } s \text{ mit } (t,s) \in R \text{ gilt : } \mathcal{A}(F,s) = 1 \\
&\equiv \text{ es gibt } s \text{ mit } (t,s) \in R : \ (\mathcal{A}(F,s) \neq 1) \\
&\equiv \text{ es gibt } s \text{ mit } (t,s) \in R : \ \mathcal{A}(\neg F, s) = 1 \\
&\equiv \mathcal{A}(\Diamond\neg F, t) = 1.
\end{aligned}$$

Mehr dazu auf den Übungsblättern.

Zur Erinnerung: aufgrund der Definitionen von $\Diamond$ und $\Box$ hängt der Wahrheitswert einer Formel F ab von

- dem Rahmen (W, R), in dem wir F betrachten,
- dem jeweiligen Referenzpunkt $s \in W$, in dem wir $\mathcal{A}(F, s)$ auswerten,

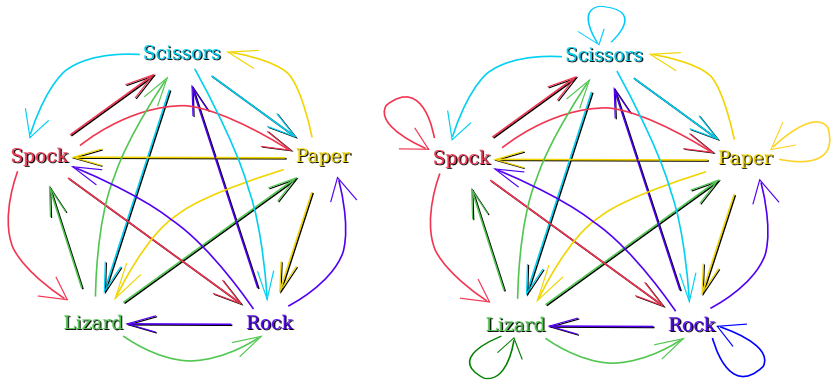
- sowie von allen Punkten, die von s aus erreichbar sind,
- und von α natürlich.

Nehmen wir zunächst an, wir hätten einen Rahmen (W,R) festgelegt. Das folgende Ergebnis beantwortet die Frage, wie weit wir von s aus gucken müssen, um "wahr" oder "falsch" entscheiden zu können. Die folgenden Definitionen ermöglichen es uns, dieses "weit" zu messen in W bezüglich F.

Definition 3.10. Sei F eine Formel und (W,R) ein Rahmen für F.
Ein Punkt $t \in W$ ist **erreichbar von** s **in** n **Schritten**, falls es $t_0,t_1,\ldots,t_n \in W$ gibt, so dass $s = t_0$, $t = t_n$ und $(t_i,t_{i+1}) \in R$ für $i = 0,1,\ldots,n-1$. Die n-te Iteration von R ist

$$R^n := \{(s,t) \in W \times W \mid t \text{ ist erreichbar von } s \text{ in } \leq n \text{ Schritten.}\}$$

Beispiel 3.11. (Beispiel 3.4 Fortsetzung.) Die Graphen von (W,R^2) und (W,R^3) sehen wie folgt aus (sie sind fast identisch: für alle unterschiedlichen Punkte $s \neq t$ gilt, dass t in zwei Schritten von a aus zu erreichen ist. Nur ein Pfad von einem Knoten s zurück zu s selbst erfordert 3 Schritte):



Beachten Sie, dass die Graphen für (W,R^n) im ersten Beispiel alle gleich dem Graph von (W,R) sind (da die Relation $<$ *transitiv* ist). Weitere Beispiele finden Sie auf den Übungsblättern.

Definition 3.12. Sei F eine Formel. Der **modale Rang** MR(F) von F ist induktiv definiert wie folgt.

- Wenn F eine atomare Formel ist, dann ist MR(F)=0.
- Wenn $F = \neg G$, dann MR(F)=MR(G).
- Wenn $F = G \wedge H$ oder $F = G \vee H$, dann MR(F)=max$\{$MR(G),MR(H)$\}$.
- Wenn $F = \Box G$ oder $F = \Diamond G$, dann MR(F)=MR(G)+1.

Zum Beispiel ist für $F = \Diamond(\Diamond A \Rightarrow \Diamond B) \Leftrightarrow \Box A$ der modale Rang MR$(F) = 2$.

Satz 3.13 (Koinzidenzlemma)**.** *Sei F eine Formel mit MR$(F) = m$, sei (W,R) ein Rahmen für F und sei $s \in W$. Weiterhin seien $\mathcal{A} = (W,R,\alpha)$ und $\mathcal{A}' = (W,R,\beta)$ zwei Strukturen für F, die auf allen t identisch sind, die s aus in höchstens m Schritten erreicht werden können. Dann ist $\mathcal{A}(F,s) = 1$ genau dann, wenn $\mathcal{A}'(F,s) = 1$.*

Im Klartext bedeutet dies: Ob eine Formel F unter $\mathcal{A}$ im Punkt s wahr ist hängt nur vom Wert von $\mathcal{A}$ in jenen Punkten $t \in W$ ab, die von s aus in höchstens $\mathrm{MR}(F)$-Schritten erreichbar sind.

Bezogen auf das obige Beispiel 3.5 bedeutet das: wenn wir den Wert von $\mathcal{A}(A,s)$ bestimmen wollen, d. h. $\alpha(A,s)$, dann müssen wir alle Punkte im Diagramm berücksichtigen, die in höchstens zwei Schritten erreichbar sind. Bedauerlicherweise bedeutet das hier: wir müssen alle Jahre in der Zukunft beachten, da ja jedes Jahr in der Zukunft in nur einem Schritt erreichbar ist: $(n,m) \in R$ immer dann, wenn $n < m$.

Bemerkung 3.14. Auch in der Modallogik gibt es Normalformen für Formeln. Das Vorgehen ist analog zum Algorithmus 1.18, zusammen mit dem Ziehen von $\Box$s und $\diamond$s direkt vor die Literale (analog zu dem Ziehen der $\neg$ vor die Literale). Da wir jedoch hier nicht über beide Analoga der deMorganschen Regeln verfügen, sondern jeweils nur über eines (siehe Satz 3.9 Regeln 6. und 7.) gibt es keine saubere normale Form, nur eine "quasi" KNF. Die besteht im Allgemeinen aus ineinander verschachtelten KNFs (eine KNF innerhalb einer KNF innerhalb..., mehr dazu siehe im Buch von Kreuzer-Kühling). Wir führen das daher hier nicht aus, denn wir benötigen im Folgenden keine Normalformen, da wir den Tableaukalkül verwenden werden.

3.3 Tableaukalkül für Modallogik

Im Folgenden werden wir den Tableau-Kalkül aus Abschnitt 1.8 auf die Modallogik erweitern. Zuvor stellen wir das Hauptergebnis dieses Abschnitts dar. Im Klartext besagt es, dass die Erfüllbarkeit von Formeln in der Modallogik immer entscheidbar ist.

Satz 3.15. *Eine Formel F in der Modallogik ist genau dann unerfüllbar, wenn das fertige Tableau geschlossen ist. Die Anzahl der Eckpunkte in jedem Tableau für F beträgt höchstens $O(m)$, wobei m die Anzahl der Teilformeln von F ist.*

Für einen Beweis siehe Kreuzer-Kühling.

Beachten Sie, dass die Anzahl der Teilformeln einer Formel der Länge n (n Symbole jeglicher Art, also etwa $\neg$, $\Box$, A, $\vee$...) im schlimmsten Fall $O(n)$ beträgt.

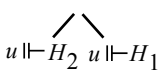
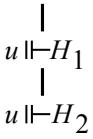
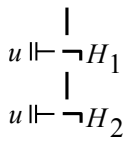
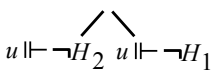
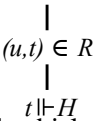
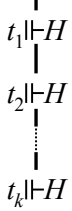
Der folgende Algorithmus ist eine Erweiterung von 1.41. Der Vollständigkeit halber listen wir hier alle alten und neuen Regeln auf. Das ist der Punkt, an dem die Notation $s \Vdash F$ praktisch ist. (Das ist ja die Abkürzung für $\mathcal{A}(F,s) = 1$.)

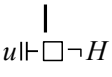
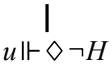
Im Folgenden heißt "Pfad" immer ein Pfad nach unten (zu den Blättern hin). entweder vom aktuellen Knoten oder von der Wurzel aus.

Algorithmus 3.1 (Tableaurechnung für Modallogik) Eingabe: eine Formel F in Modallogik.

1. Starte mit $s \Vdash F$ als Wurzel.

2. Wähle einen unmarkierten Knoten $u \Vdash G$, wobei G kein Literal ist (im Sinne der Aussagenlogik). Markiere $u \Vdash G$. Wende die folgenden Regeln an, bis alle Möglichkeiten ausgeschöpft sind.

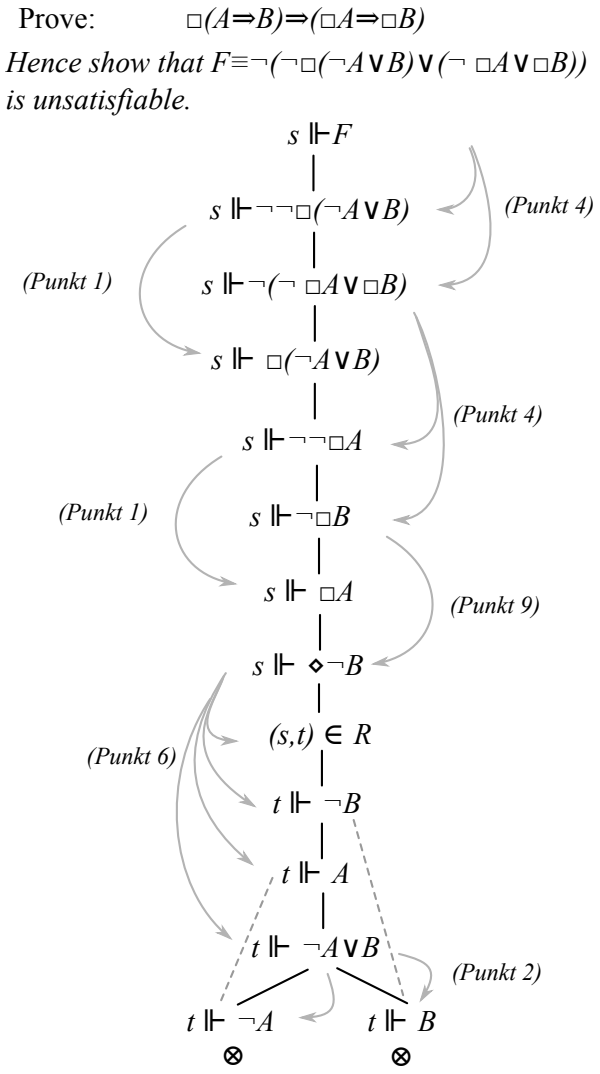
   - Wenn G die Form $u \Vdash \neg\neg H$ hat, dann füge einen einzelnen Knoten $u \Vdash H$ zu jedem Pfad hinzu, der in $u \Vdash G$ beginnt.
   - Wenn G die Form $H_1 \vee H_2$ hat, dann füge  zu jedem Pfad hinzu, der in $u \Vdash G$ beginnt.
   - Wenn G die Form $H_1 \wedge H_2$ hat, dann füge  zu jedem Pfad ninzu, der in $u \Vdash G$ beginnt.
   - Wenn G die Form $\neg(H_1 \vee H_2)$ hat, dann füge  zu jedem Pfad hinzu, der in $u \Vdash G$ beginnt.
   - Wenn G die Form $\neg(H_1 \wedge H_2)$ hat, dann füge  zu jedem Pfad hinzu, der in $u \Vdash G$ beginnt.
   - Wenn G die Form $\diamond H$ hat, dann füge  zu jedem Pfad hinzu, der in $u \Vdash G$ beginnt, wobei $t \in W$ ein Symbol ist, das bislang noch nicht im Tableau vorkommt. Für jeden Knoten $u \Vdash \Box H'$ in dem Pfad von der Wurzel durch $u \Vdash G$ füge  zu jedem Pfad hinzu, der in $u \Vdash \Box H'$ beginnt und $u \Vdash G$ enthält.
   - Wenn G die Form $\Box H$ hat, dann füge  zu jedem Pfad hinzu, der in $u \Vdash G$ beginnt.

     Dabei sind $t_1, \ldots, t_k$ die Punkte, die in einem Knoten der Form $(u, t_i) \in R$ auftreten, der in diesem Pfad enthalten ist.
   - Wenn G die Form $\neg\diamond H$ hat, dann füge  zu jedem Pfad hinzu, der in $u \Vdash G$ beginnt.
   - Wenn G die Form $\neg\Box H$ hat, dann füge  zu jedem Pfad hinzu, der in $u \Vdash G$ beginnt.

3. Wenn es keinen weiteren Knoten zum Markieren gibt, geben Sie das Tableau zurück, STOP.

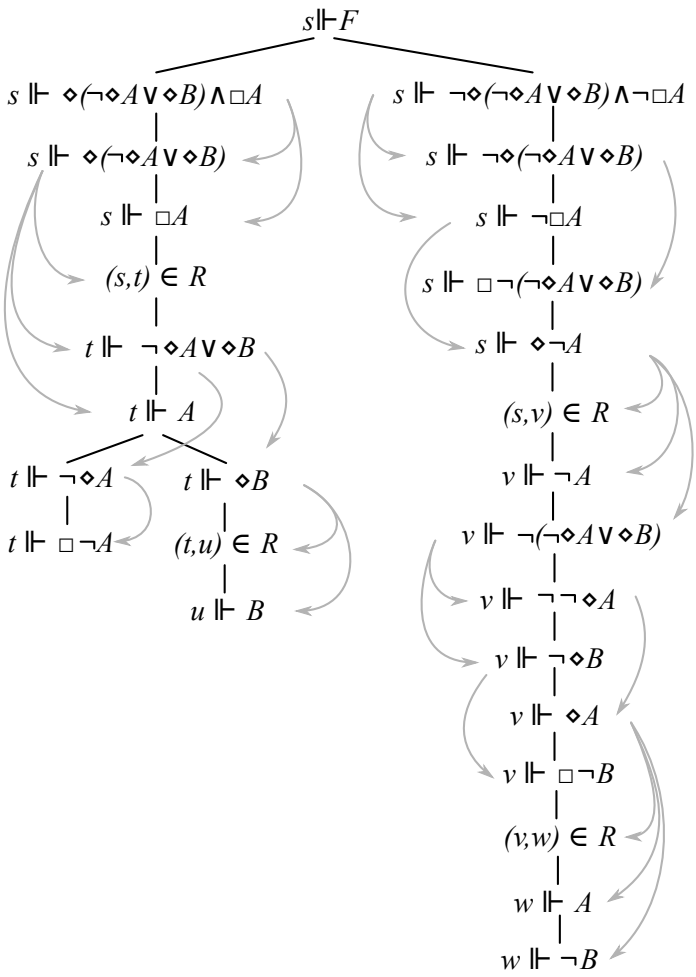
Ein Pfad ist **geschlossen**, wenn er $u \Vdash A_i$ und $u \Vdash \neg A_i$ enthält für ein paar i. In diesem Fall markieren wir das Blatt dieses Pfades mit $\otimes$. A Tableau ist **geschlossen**, wenn alle Blätter mit $\otimes$ markiert sind. In diesem Fall ist F unerfüllbar. Sonst ist F erfüllbar.

Als nächstes zeigen wir zwei Beispiele: Zuerst beweisen wir Regel 3 des Satzes 3.9. indem wir zeigen, dass $\Box(F \Rightarrow G) \Rightarrow (\Box F \Rightarrow \Box G)$ eine Tautologie ist; also indem wir zeigen, dass dessen Negation $\neg\big(\Box(F \Rightarrow G) \Rightarrow (\Box F \Rightarrow \Box G)\big)$ unerfüllbar ist. Dann zeigen wir, dass unser Meisterbeispiel $\diamond(\diamond A \Rightarrow \diamond B) \Leftrightarrow \Box A$ erfüllbar ist.



Example: $F \equiv \diamond(\diamond A \Rightarrow \diamond B) \Leftrightarrow \Box A$

$\equiv (\diamond(\neg\diamond A \vee \diamond B) \wedge \Box A) \vee (\neg\diamond(\neg\diamond A \vee \diamond B) \wedge \neg\Box A)$



3.4 Varianten der Modallogik

9. Jan

Bislang können als Rahmen alle möglichen Relationen gewählt werden. Viel Arbeit floss in das Studium der Modallogik, bei dem die Relationen bestimmte Eigenschaften haben. Drei die wir schon kennen sind:

- $\forall s \in W : (s,s) \in R$ (reflexiv)
- $\forall s,t \in W : (s,t) \in R \Rightarrow (t,s) \in R$ (symmetrisch)
- $\forall s,t,u \in W : \big((s,t) \in R \wedge (t,u) \in R\big) \Rightarrow (s,u) \in R$ (transitive)

Falls wir fordern, dass unser Rahmen ein oder mehrere dieser Eigenschaften hat, dann gibt es plotzlich weitere Rechenregeln, die in der allgemeinen Modallogik nicht wahr sind. Z.B. ist $F \models diamondF$ wahr, falls (W,R) reflexiv ist (vgl. auch die Übungen).

$$s \Vdash A \Rightarrow \diamond A \equiv s \Vdash \neg A \vee s \Vdash \diamond A \equiv s \Vdash \neg A \vee \exists t \text{ accessible from } s : t \Vdash A,$$

und mit $s = t$ ($(s,s) \in R$!) wird daraus $s \Vdash \neg A \vee s \Vdash A$. Das ist offenbar eine Tautologie.

In der Literatur wurden insbesondere die folgenden Varianten der Modallogik untersucht.

K	Keine weiteren Bedingungen an (W,R)
D	seriell, d.h. $\forall s \in W \exists t \in W : (s,t) \in R$ (keine Sackgassen)
T	reflexiv
S4	reflexiv und transitiv
S5	reflexiv, symmetrisch und transitiv

Es ist leicht zu sehen, dass für diese fünf Punkte eine Hierarchie gilt: jede der Logiken enthält alle darunter. (Weil z.B. jede Relation,, die reflexiv *und* transistiv ist, auch reflexiv ist; bzw. weil aus reflexiv seriell folgt.)

Eine weitere wichtige Variante der Modallogik nutzt die Zeit als Rahmen. Das ist **temporale Logik**: Sie verwendet die gleichen Elemente wie die modale Logik, aber für den Rahmen (W,R) verlangt man zusätzlich, dass er transitiv ist und *irreflexiv*: $\forall s \in W : (s,s) \notin R$. So realisiert etwa ein Rahmen $(\mathbb{N}_0, <)$ temporale Logik.

Der Tableaukalkül von oben (also auch Satz 3.15) gilt für die modale Logik K. Für die anderen braucht man ein Tablezukalkül mit mehr Regeln; oder aber man bekommt keinen perfekten Tst mehr, sondern nur "wenn alle Pfade geschlossen, dann unerfüllbar".

Es ist bekannt, dass die Modallogiken K, T, S4, S5 und die temporale Logik alle entscheidbar sind, auch wenn der Tableau-Algorithmus in der obigen Form möglicherweise in allen Fällen terminiert (z. B. in der temporalen Logik).

Zwischenspiel: Unendliche Kardinalitäten

Für eine endliche Menge M ist klar, wie die Anzahl ihrer Elemente zu definieren ist. Diese Zahl wird als **Kardinalität** von M bezeichnet und mit $|M|$ bezeichnet (manchmal auch $\#M$ oder $\text{card}(M)$). Wenn M unendlich viele Elemente hat das wird schwieriger. Dann hilft folgendes Konzept:

Definition 3.16. Seien M,N zwei Mengen. Es gilt $|M| = |N|$ (Sprechweise: M und N haben die gleiche **Kardinalität**, falls es eine bijektive Abbildung von M nach N gibt.
Es gilt $|M| \leq |N|$, falls es falls es eine injektive Abbildung von M nach N gibt.

Eine Abbildung $f : M \to N$ heißt *bijektiv*, wenn es für jedes $m \in M$ genau ein $n \in \mathbb{N}$ gibt, so dass $f(m) = n$. Stellen Sie sich eine Massenhochzeit vor: M besteht aus einer Gruppe von Männern, N besteht aus einer Gruppe von Frauen. Dann ordnet ein bijektives f jedem Mann genau eine Frau zu und umgekehrt. Für endliche Mengen M und N ist dies offensichtlich nur dann möglich, wenn sie die gleiche Anzahl an Elementen haben.

Damit können wir den Begriff der Kardinalität nun auf unendliche Mengen erweitern, etwa $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{Q}$, $\mathbb{R}$... oder auf ihre Potenzmengen.

Definition 3.17. Die **Potenzmenge** einer Menge M ist die Menge aller ihrer Teilmengen, einschließlich M und $\varnothing$. Es wird mit $\mathcal{P}(M)$ bezeichnet.

Zum Beispiel ist die Potenzmenge von $\{1,2,3\}$

$$\mathcal{P}(\{1,2,3\}) = \{\varnothing, \{1\}, \{2\}, \{3\}, \{1,2\}, \{1,3\}, \{2,3\}, \{1,2,3\}\}$$

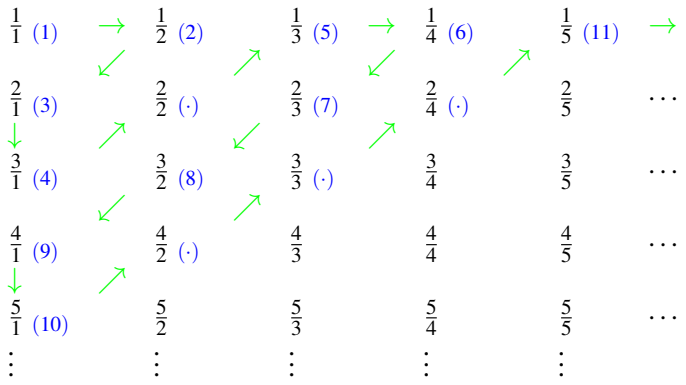
Die erste vielleicht überraschende Tatsache ist die folgende. (Überraschend, denn für Bei endlichen Mengen ist es nicht möglich, dass eine echte Teilmenge von M dasselbe hat Kardinalität als M.)

Satz 3.18. $|\mathbb{N}| = |\mathbb{Q}|$.

Proof. Wir müssen eine bijektive Abbildung f von $\mathbb{Q}$ nach $\mathbb{N}$ erstellen. Schreiben Sie alles Elemente von $\mathbb{Q}$ in einem unendlichen zweidimensionalen Array:

$$\begin{array}{ccccccc}
\frac{1}{1} & \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \cdots \\
\frac{2}{1} & \frac{2}{2} & \frac{2}{3} & \frac{2}{4} & \frac{2}{5} & \cdots \\
\frac{3}{1} & \frac{3}{2} & \frac{3}{3} & \frac{3}{4} & \frac{3}{5} & \cdots \\
\frac{4}{1} & \frac{4}{2} & \frac{4}{3} & \frac{4}{4} & \frac{4}{5} & \cdots \\
\frac{5}{1} & \frac{5}{2} & \frac{5}{3} & \frac{5}{4} & \frac{5}{5} & \cdots \\
\vdots & \vdots & \vdots & \vdots & \vdots
\end{array}$$

Offenbar enthält dieses Array jedes positive Element von $\mathbb{Q}$. Manche von ihnen kommen mehrmals vor (z. B. $\frac{1}{1} = \frac{2}{2} = \frac{3}{3} = \cdots$) Gehen wir nun durch dieses Array, wie hier im folgenden Diagramm gezeigt: Nummeriere jedes Element von $\mathbb{Q}$, dass nicht bereits erfasst ist, nacheinander mit den natürlichen Zahlen. (z. B. $\frac{2}{2}, \frac{3}{3}, \ldots$ nicht markieren, da $\frac{1}{1}$ bereits markiert ist).



Es ist klar, dass jede Zahl in $\mathbb{Q}$ in diesem Array markiert wird: Sie wird erreicht nach endlich vielen Schritten. Dies ergibt eine Bijektion zwischen $\mathbb{N}$ und den positiven rationalen Zahlen $\mathbb{Q}^+$:

$$\begin{array}{cccccccccccc}
1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & \cdots \\
\downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\
1 & \frac{1}{2} & 2 & 3 & \frac{1}{3} & \frac{1}{4} & \frac{2}{3} & \frac{3}{2} & 4 & 5 & \frac{1}{5} & \cdots
\end{array}$$

Der Rest ist einfach: Füge am Anfang die 0 ein, und hinter jeden positiven rationalen Zahl ihr negatives:

$$\begin{array}{cccccccccccccccc}
1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & \cdots \\
\downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\
0 & 1 & -1 & \frac{1}{2} & -\frac{1}{2} & 2 & -2 & 3 & -3 & \frac{1}{3} & -\frac{1}{3} & \frac{1}{4} & -\frac{1}{4} & \frac{2}{3} & -\frac{2}{3} & \cdots
\end{array}.$$

Dies ist die gewünschte bijektive Abbildung, daher $|\mathbb{N}| = |\mathbb{Q}|$. □

Dieser Beweis zeigt einen allgemeinen Trick: Um $M| = |\mathbb{N}|$ zu zeigen, reicht es aus, alle Elemente in M mit $1,2,3,\ldots$ so zu nummerieren, so dass jedes Element in M genau eine Zahl $n \in \mathbb{N}$ bekommt. Eine zweite, etwas überraschende Tatsache ist diese:

Satz 3.19 (Cantor 1874)**.** $|\mathbb{R}| > |\mathbb{N}|$.

Der folgende Beweis ist nicht das Original, sondern einer von Cantor aus dem Jahr 1891.

Proof. durch Widerspruch. Wir zeigen, dass es keine Bijektion von $\mathbb{N}$ in das halboffene Intervall $[0;1[$ gibt (d.h. alle Zahlen x, so dass $0 \leq x < 1$). Wenn es uns das gelingt, zeigt das, dass es erst recht keine Bijektion gibt zwischen $\mathbb{R}$ und $\mathbb{N}$, da ja $\mathbb{R}$ noch größer als $[0,1[$ ist.

Nehmen wir also an, dass es eine Bijektion von $\mathbb{N}$ in $[0;1[$ gibt. Dann können wir die Elemente $a_1, a_2, \ldots$ von $[0;1[$ in eine Liste schreiben. Wir verwenden die dezimale Darstellung $0.a_{i,1}a_{i,2}a_{i,3}\ldots$.

$$\begin{array}{llllllll}
a_1 = 0. & a_{1,1} & a_{1,2} & a_{1,3} & a_{1,4} & a_{1,5} & \ldots \\
a_2 = 0. & a_{2,1} & a_{2,2} & a_{2,3} & a_{2,4} & a_{2,5} & \ldots \\
a_3 = 0. & a_{3,1} & a_{3,2} & a_{3,3} & a_{3,4} & a_{3,5} & \ldots \\
a_4 = 0. & a_{4,1} & a_{4,2} & a_{4,3} & a_{4,4} & a_{4,5} & \ldots \\
a_5 = 0. & a_{5,1} & a_{5,2} & a_{5,3} & a_{5,4} & a_{5,5} & \ldots \\
 & & & \ldots & & &
\end{array}$$

Wenn es eine Mehrdeutigkeit gibt (wie $0,09999\cdots = 0,1$), wählen wir die endliche Version, die auf ...0000... endet, und nicht die unendliche Version, die mit ...99999... endet.

Nach unserer Annahme enthält diese Liste alle Elemente von $[0,1[$. Wir werden nun einen Widerspruch finden, indem wir ein Element konstruieren, das nicht in der Liste enthalten ist: definiere $s = 0.s_1s_2s_3\cdots$ by

$$s_i = \begin{cases} 7 & \text{if } a_{ii} = 3 \\ 3 & \text{if } a_{ii} \neq 3 \end{cases}$$

Diese Zahl s liegt auf jeden Fall in $[0,1[$ und endet nicht mit ...99999... (da ihre Ziffern alle entweder 3 oder 7 sind). Die erste Ziffer von s nach dem Punkt unterscheidet sich von der ersten Ziffer von a_1 (konstruktionsbedingt: wenn die erste Ziffer von a_1 eine 3 ist, dann ist die erste Ziffer von s eine 7; und wenn die erste Ziffer von a_1 keine 3 ist, dann ist die erste Ziffer von s eine 3). Ganz analog sieht man, dass die i-te Ziffer von s hinter dem Komma ungleich ist der i-ten Ziffer von a_i.

Daher ist $s \neq a_i$ für alle i, also für alle Ziffern in der Liste. □

Das letzte Ergebnis führt zu der folgenden Terminologie.

Definition 3.20. Eine Menge M mit $|M| = |\mathbb{N}|$ heißt **abzählbare** Menge. Eine Menge M mit $|M| > |\mathbb{N}|$ heißt **überabzählbare** Menge.

Da wir nun wissen, dass es verschiedene Arten von "unendlich vielen" gibt: Können wir diese Hierarchie beschreiben? Auch hier hilft Georg Cantor.

Satz 3.21. *Sei M eine Menge. Dann ist $|\mathcal{P}(M)| > |M|$.*

Für eine endliche Menge $M = \{m_1, m_2, \dots, \}$ ist dies eine einfache Beobachtung: $\mathcal{P}(M)$ enthält mindestens $\{m_1\}, \{m_2\}, \dots$, aber auch $\{m_1, m_2\}$, oder M selbst. Beachten Sie, dass $|\varnothing| = 0$, aber

$$|\mathcal{P}(\varnothing)| = |\{\varnothing\}| = 1.$$

Insbesondere impliziert der Satz $|\mathcal{P}(\mathbb{N})| > |\mathbb{N}|$, $|\mathcal{P}(\mathcal{P}(\mathbb{N}))| > |\mathcal{P}(\mathbb{N})|$ usw. Für diese wachsenden unendlichen Kardinalitäten gibt es eine Notation.

Definition 3.22. Sei $\beth_0 := |\mathbb{N}|$. Falls $|M| = \beth_n$ für ein $n \geq 0$, dann sei $\beth_{n+1} := |\mathcal{P}(M)|$.

Bemerkung 3.23. Man kann zeigen, dass $|\mathbb{R}| = |\mathcal{P}(\mathbb{N})| = \beth_1$.

Normalerweise ist der einfachste Weg, solche Gleichheiten zu zeigen, das Schröder-Bernstein-Theorem:

Satz 3.24. *Seien M, N zwei Mengen. Gibt es injektive Funktionen $f : M \to N$ und $g : N \to M$ zwischen den Mengen A und B, dann gilt $|A| = |B|$.*

Das $\beth$ (”beth”) ist der zweite Buchstabe im hebräischen Alphabet. Es ist (war) eine große Frage, ob diese $\beth_n$s die *einzigen* Unendlichkeiten sind, oder ob dazwischen noch mehr unendliche Zahlen liegen. Insbesondere fragt die Kontinuumshypothese ob es eine unendliche Kardinalität zwischen $\beth_0$ und $\beth_1$ gibt. Klicken Sie auf den Link für weitere Informationen (Wikipedia). Der Wikipedia-Artikel verwendet Begriffe wie ”unabhängig von ZFC” usw. Diese werden im Abschnitt 5 erklärt.

4 Unentscheidbarkeit

16. Jan

Ein gefeiertes Ergebnis im Bereich der Unentscheidbarkeit ist der legendären (und tragischen) Figur von Kurt Gödel zu verdanken. Es gibt zwei unterschiedliche Bedeutungen des Wort ”unentscheidbar” im Kontext der Logik oder Berechenbarkeit. Die erste davon wird in Bezug auf die Berechenbarkeitstheorie verwendet. Sie gilt nicht für einzelne Aussagen (”dieser Satz ist wahr”), sondern für Entscheidungsprobleme, die (abzählbar) *unendlich viele* Eingaben haben (Z.B. ”Ist n eine Primzahl?” für $n \in \mathbb{N}$), die jeweils mit Ja oder Nein beantwortet werden müssen. Genauer: wir nennen ein solches Problem **nicht berechenbar**, wenn es kein Computerprogramm gibt (genauer: keine Turingmaschine), das die Frage für jedes n richtig mit “Ja” oder “Nein” beantwortet.

Die zweite Bedeutung dieses Begriffs ist die Bedeutung, die in Bezug auf die Gödelschen Sätze verwendet wird: dass eine *einzelne* Aussage in einem gegebenen logischen System weder beweisbar noch widerlegbar ist. Hier verwenden wir das Wort **nicht entscheidbar**.

4.1 Unberechenbare Probleme

Definition 4.1. Ein **Entscheidungsproblem** ist eine Ja-Nein-Frage mit unendlich vielen möglichen Eingaben. Formal kann es als Paar (I, M) modelliert werden, wobei I die Menge aller möglichen Eingaben ist und $M \subset I$ ist die Menge der Eingaben mit Antwort ”Ja”.

Ein Problem ist **berechenbar** (auch bekannt als computable, entscheidbar) wenn es eine Turing-Maschine gibt, die auf jede Eingabe korrekt mit ”Ja” oder ”Nein” antwortet. (Wenn Sie nicht wissen, was eine Turingmaschine ist, ersetzen Sie ”Turingmaschine” durch ”Algorithmus” und denken Sie an einen Haskell- oder Python-Algorithmus.) Ein Problem ist **semi-berechenbar** (auch bekannt

als semi-computable), falls es eine Turingmaschine gibt, die bei jeder Eingabe, für die die korrekte Antwort "Ja" lautet, auch mit "Ja" antwortet, und bei keiner Eingabe, für die die korrekte Antwort "Nein" lautet, auch mit "Ja" antwortet.

Normalerweise können die Eingaben aufgezählt werden. Also ist oft $I = \mathbb{N}$ (oder $I = \mathbb{N}^k$) und $M \subset \mathbb{N}$ (oder $M \subset \mathbb{N}^k$). Dann kann man die Frage modellieren als eine Funktion $f : \mathbb{N}^k \to \{0,1\}$, mit $f(n) = 1$, wenn $n \in M$, $f(n) = 0$ sonst. Die entsprechenden Namen für f sind dann **rekursiv** (korr. zu berechenbar) und **rekursiv aufzählbar** (entspricht semi-berechenbar).

Ein berühmtes unberechenbares Problem ist die Frage: "Wird Turingmaschine Nummer i jemals bei Eingabe j anhalten?" Diese Frage ist bekannt als das Halteproblem. Da es abzählbar unendlich viele Turingmaschinen gibt, können wir alle Turing-Maschinen mit Zahlen in $\mathbb{N}$ durchnummerieren. Alle möglichen Eingaben (endliche Startkonfigurationen auf dem Band) werden ebenfalls aufgezählt (nach einem geeigneten Schema).

Satz 4.2 (Turing 1937)**.** *Das Halteproblem ist unberechenbar.*

Proof. Als Funktion formuliert, fragt das Problem nun nach einer *berechbaren* Funktion $f : \mathbb{N} \times \mathbb{N} \to \{0,1\}$ wobei $f(i,j) = 1$, wenn die Turing-Maschine i bei Eingabe j stoppt, und $f(i,j) = 0$ sonst. Nehmen Sie an, dass f berechenbar ist. Wir betrachten

$$g : \mathbb{N} \to \{0,1,\text{undefiniert}\}, \quad g(i) = \begin{cases} 0 & \text{if } f(i,i) = 0 \\ \text{undefiniert} & \text{sonst} \end{cases}$$

Beispielsweise kann g so als Turingmaschine (bzw. als Programm) realisiert werden:

Wenn $f(i,i) = 0$, dann 0 zurückgeben, andernfalls ewig weiterlaufen

Offensichtlich ist g berechenbar, wenn f es ist. Es gibt also eine Turingmaschine t_g, die g berechnet. Was ist $f(t_g, t_g)$?

Fall 0: Wenn $f(t_g, t_g) = 0$, dann $g(t_g) = 0$. Das heißt also (weil die Turingmaschine t_g das g berechnet), dass t_g bei der Eingabe von t_g stoppt. Also gilt $f(t_g, t_g) = 1$. Widerspruch.

Fall 1: Wenn $f(t_g, t_g) = 1$, dann ist $g(t_g)$ nicht definiert, daher läuft t_g ewig weiter. Also ist $f(t_g, t_g) = 0$. Widerspruch.

Die Annahme, dass f berechenbar ist, führt in allen (beiden) Fällen zu einem Widerspruch, daher ist f nicht berechenbar. □

Satz 4.3 (Turing 1937)**.** *Das Halteproblem ist semi-berechenbar. D.h. es gibt eine Turingmaschine, die bei der Eingabe (i,j) eine 1 zurückgibt, wenn die Turing-Maschine i mit Eingabe j stoppt; und die nichts ausgibt, wenn nicht.*

Proof. Die grobe Idee ist: Lassen Sie einfach eine gegebene Turing-Maschine i mit der Eingabe j laufen. Wenn die stoppt, geben Sie 1 zurück. Ansonsten weiterlaufen. Dazu brauchen wir eine Turingmaschine, die alle möglichen Turingmaschinen simulieren kann.

Dazu muss man sich nur noch darum kümmern, in einer sinnvollen Reihenfolge nacheinander alle möglichen Turingmaschinen mit allen möglichen Eingaben laufen zu lassen. (bzw genauer: simulieren

zu lassen. Das ist die "universelle" Turingmaschine). Dazu berechnen wir (irgendwie) den k-ten Schritt für die Turing-Maschine i bei Eingabe j in dieser Reihenfolge:

$$(i,j,k) = (1,1,1),(1,1,2),(1,2,1),(2,1,1),(1,1,3),(1,2,2),(1,3,1),(2,1,2),(2,2,1),(3,1,1),(1,1,4),\ldots$$

Im Klartext: wir listen alle Tripel (i,j,k) mit $i,j,k \in \mathbb{N} \setminus \{0\}$ in aufsteigender Reihenfolge bezüglich $i+j+k$ auf, und arbeiten diese dann bezüglich der lexikografischen Reihenfolge ab. Es ist klar, dass alle Möglichkeiten in $\mathbb{N}^3$ so in endlicher Zeit (!) erreicht werden.

Unsere universelle Turingmaschine nimmt also (i,j) als Eingabe entgegen, simuliert irgendwann jeden k-ten Schritt der Turingmaschine i mit Eingabe j, und gibt 1 zurück, wenn die Turingmaschine i schließlich stoppt. □

Einige unberechenbare Probleme Die folgenden Probleme sind unberechenbar (aber alle semiberechenbar).

1. Ist eine gegebene Formel in der Logik erster Ordnung gültig? (Church 1936, Turing 1937)

Satz 4.4 (Church 1936, Turing 1937)**.** *Das Problem, ob eine Formel in Prädikatenlogik erfüllbar ist, ist unberechenbar.*
Da F genau dann unerfüllbar ist, wenn $\neg F$ gültig ist, ist die Frage, ob die Formel F in Prädikatenlogik gültig ist, ebenfalls unberechenbar.

Eine Strategie des Beweises ist die folgende (der ursprüngliche Beweis war anders):

I. Zeige: Wenn das obige Problem berechenbar ist, dann ist es auch das folgende (das *Post-Korrespondenzproblem*) berechenbar.

II. Zeige: Wenn das Post-Korrespondenzproblem berechenbar ist, dann ist das Halteproblem berechenbar.

Teil I findet sich im Buch von Schöning, Teil II im Buch von Sipser (beide in der Literaturliste am Ende dieses Skripts). Damit ergibt sich einen Widerspruch zu Satz 4.2. Daher muss die Frage nach der Erfüllbarkeit einer prädikatenlogischen Formel unberechenbar sein.

2. Das **Post-Korrespondenz-Problem (PKP)**: Die Eingabe des Problems besteht aus zwei endlichen Listen $u_1,\ldots,u_n$ und $v_1,\ldots,v_n$ von Wörtern über dem Alphabet $\{0,1\}$. Eine Lösung hierfür ist eine Folge von Indizes $i_1,i_2,\ldots,i_k$ mit $k \geq 1$, so dass

$$u_{i_1}\ldots u_{i_k} = v_{i_1}\ldots v_{i_k}.$$

Hier bedeutet u_1u_2 einfach Hintereinanderschreiben: wenn $u_1 = 10$ und $u_2 = 01$ ist, dann $u_1u_2 = 1001$. Die Entscheidungsversion des PKP-Problems ist, dann zu entscheiden ob es eine solche Lösung gibt oder nicht.

Eine schöne Veranschaulichung dieses Problems ist: Nehmen wir an, wir haben endlich viele verschiedene Typen von Dominosteine mit einigen 0-1-Wörtern oben und unten. Von jedem Typ dürfen wir eine beliebige große Anzahl von Kopien nutzen. Können wir die so (hochkant) in eine Reihe nebeneinander legen, dass in der oberen Zeile dieselbe Zeichenfolge angezeigt wird wie in der unteren Zeile?

Nehmen wir zum Beispiel an, dass $u_1 = 1, u_2 = 10, u_3 = 011$ und $v_1 = 101, v_2 = 00, v_3 = 11$. Die Dominos sehen somit wie folgt aus:

$$\left[\begin{smallmatrix}1\\101\end{smallmatrix}\right], \quad \left[\begin{smallmatrix}10\\00\end{smallmatrix}\right], \quad \left[\begin{smallmatrix}011\\11\end{smallmatrix}\right].$$

Eine Lösung ist $(1,3,2,3)$, d.h.

$$\left[\begin{smallmatrix}1\\101\end{smallmatrix}\right]\left[\begin{smallmatrix}011\\11\end{smallmatrix}\right]\left[\begin{smallmatrix}10\\00\end{smallmatrix}\right]\left[\begin{smallmatrix}011\\11\end{smallmatrix}\right].$$

Das Wort in der oberen und unteren Reihe lautet 101110011.

3. Das **Wang-Kachelproblem**: Gegeben sei eine Menge von Quadraten, deren Ränder sind mit verschiedenen Farben gefärbt sind ("Wang-Kacheln"). Können wir mit Kopien davon die Ebene so pflastern, dass aneinanderstoßende Kanten die gleiche Farbe haben? "Die Ebene pflastern" bedeutet: die Kacheln so legen, dass sie nicht überlappen und keine Lücken lassen. Die Kacheln sollten immer Ecke an Ecke liegen. Es ist nicht erlaubt, die Kacheln zu drehen oder zu spiegeln.

Mit dem unten aufgeführten Satz von 11 Kacheln kann die Ebene gepflastert werden (aber es ist sehr knifflig zu sehen, wie).

Keine Teilmenge von 10 dieser 11 Kacheln kann die Ebene pflastern. Im Allgemeinen ist dieses Problem unberechenbar für mehr als drei Farben und mehr als 10 Kacheln.

4. **Sterbliche Matrizen**: Gegeben sind Matrizen $A_1,\ldots,A_n$ in $\mathbb{Z}^{d\times d}$. Gibt es eine Kombination davon (z. B. $(i_1,i_2,\ldots,i_m)$ mit $1\le i_j\le n$), so dass das Produkt gleich der Nullmatrix ist? Das heißt, gibt es $i_1,\ldots,i_m$ so dass

$$A_{i_1}\cdot A_{i_2}\cdots A_{i_m}=\begin{pmatrix}0&\cdots&0\\\vdots&&\vdots\\0&\cdots&0\end{pmatrix}?$$

Dieses Problem ist unberechenbar für $n\ge 6$, falls $d=3$ (auch für $n\ge 15$ und $n\ge 2$).

5. **Conways Spiel des Lebens** (siehe Wikipedia): für ein gegebenes Startmuster und ein anderes Muster, kann letzteres jemals das Ergebnis des ersteren sein?

6. **Diophantische Gleichungen:** Gegeben sei ein Polynom mit mehreren Variablen, aber mit ganzzahlige Koeffizienten. Gibt es eine ganzzahlige Lösung? (diophantisch bedeutet, dass wir nur an ganzzahligen Lösungen interessiert sind). Nehmen wir zum Beispiel die die diophantische Gleichung $3x^2-2xy-y^2z-7=0$. Die hat eine ganzzahlige Lösung: $x=1,\ y=2,\ z=-2$. Im Gegensatz dazu hat die diophantische Gleichung $x^2+4y^2-3=0$ keine solche Lösung (sondern nur nicht ganzzahlige, wie $x=1,y=\frac{\sqrt{2}}{2}$, oder $x=\sqrt{2},y=\frac{1}{2}$).

Bei allen kann gezeigt werden, dass sie semi-berechenbar sind (aber nicht berechenbar), indem sie (direkt oder indirekt) auf das Halteproblem zurückgeführt werden.

4.2 Berechenbare Zahlen

In dem Artikel, in dem Turing den Satz 4.4 bewies, wurden ebenfalls das Konzept der Turingmaschine eingeführt, auch die universelle Turingmaschine. Dar Artikel trug den Titel "On computable numbers with an application to the Entscheidungsproblem". (Ja, so. Das ist kein Google-Translate-Artefakt.) Turing definierte eine Zahl als berechenbar wie folgt: alle ganzen Zahlen sind berechenbar. Eine Zahl $x\notin\mathbb{Z}$ ist berechenbar, wenn es eine Turing-Maschine gibt, die bei Eingabe n die n-te Ziffer ihrer Dezimaldarstellung berechnet ("die n-te Nachkommastelle"). Das entspricht fast der modernen Definition:

Definition 4.5. Eine Zahl $x \in \mathbb{R}$ ist **berechenbar**, wenn für jedes $\varepsilon \in \mathbb{Q}, \varepsilon > 0$ Es gibt eine berechenbare Funktion $f : \mathbb{Q} \to \mathbb{Q}$ mit $|x - f(\varepsilon)| < \varepsilon$.
Die Menge aller berechenbaren Zahlen wird mit $\mathbb{B}$ bezeichnet.

Die moderne Definition löst das folgende Problem: eine Turingmaschine, die die Zahl 2 berechnen soll, kann das ja tun, indem sie $1,9999999\ldots$ berechnet. Es gilt ja $2 = 1,999999\ldots$. Aber die siebte Nachkommastelle von 2 ist ja 0, nicht 9. Die Definition oben repariert dieses Problem.

Turing erkannte bereits, dass nicht alle Zahlen berechenbar sind. Das Argument ist einfach: Es gibt nur abzählbar viele Turingmaschinen (weil jede Turing-Maschine als ein endliches 0-1-Wort codiert werden kann, gibt es höchstens abzählbar viele. Also auch maximal abzählbar viele berechenbare Zahlen. Da $\mathbb{R}$ ist überabzählbar ist gilt:

Fast alle Zahlen in $\mathbb{R}$ sind nicht berechenbar.

Trotzdem, Turing hat bereits gezeigt:

1. Alle rationalen Zahlen sind berechenbar. Mit anderen Worten/Symbolen: $\mathbb{Q} \subset \mathbb{B}$
2. Alle algebraischen Zahlen sind berechenbar (also die Wurzeln eines Polynoms). mit ganzzahligen Koeffizienten).
3. π und e sind berechenbar.
4. Der Grenzwert einer berechenbaren Folge, die berechenbar konvergent ist ist berechenbar.

Um Punkt 4 zu erklären: Es stimmt nicht, dass der Grenzwert irgendeiner konvergenten Folge von berechenbaren Zahl berechenbar ist. Wir müssen sicherstellen, dass: (a) die Sequenz selbst berechenbar ist und (b) die Tatsache, *dass* sie konvergiert, berechenbar ist.

Definition 4.6. Eine Folge berechenbarer Zahlen a_n heißt **berechenbar konvergent**, wenn es eine berechenbare Funktion $N : \mathbb{B} \to \mathbb{N}$ gibt, so dass für alle $\varepsilon \in \mathbb{B}, \varepsilon > 0$ und für alle $m, n \in \mathbb{N}$, so dass $m > N(\varepsilon)$ und $n > N(\varepsilon)$ gilt $|a_n - a_m| < \varepsilon$.

Vergleichen Sie diese Definition mit der Definition von a Cauchy-Sequenz.

Das erledigt nun Punkt 4 von oben:

$$\pi = 4(1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \cdots) \quad \text{und} \quad e = 1 + \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \frac{1}{4!} + \cdots$$

sind berechenbar. Darüber hinaus stellt Punkt 4 sicher, dass alle algebraischen Zahlen berechenbar sind, da es effiziente Methoden zur Approximation von Nullstellen von Polynomen mit ganzzahligen Koeffizienten gibt. Tatsächlich sind praktisch *alle* Zahlen, die wir mit Worten beschreiben können, berechenbar. Ausnahmen sind wie folgt aufgebaut:

Eine nicht-berechenbare Zahl Betrachten wir wieder das Halteproblem für Turing-Maschinen. Nehmen wir an, wir haben uns bereits auf eine Aufzählung aller Turingmaschinen geeinigt. Sei $0 < a < 1$, sodass die n-te Nachkommastelle von a eine 0 ist, falls Turingmaschine Nummer n bei Eingabe 0 irgendwann anhält, und 1 sonst. Es ist klar, dass $a \in \mathbb{R}$. Wenn a berechenbar wäre, hätten wir eine Lösung für das Halteproblem. Dies widerspricht dem Satz 4.3.

4.3 Folgerungen II

Wir sind mittlerweile in der Situation, Gödels Ergebnisse genau verstehen zu können (und nicht nur mittels oberflächlicher Veranschaulichungen). Wir erinnern nochmal an die beiden Arten von Folgerung, fassen sie nun aber in einem anderen Rahmen auf (nämlich, es gibt ein paar Formeln, die wir als wahr auffassen). 23. Jan

Definition 4.7. Ein **formales System** ist ein Paar $(\mathcal{F}, S)$, wobei $\mathcal{F}$ eine Menge von Formeln ist (die **Axiome**), von denen wir annehmen, dass sie wahr sind; sowie einen Satz S von Regeln zum ziehen syntaktischer Folgerungen (z.B. Resolutionskalkül, oder Modus Ponens, oder beides zusammen).

Eine Formel G ist eine **semantische Folgerung** von F, wenn für alle $\mathcal{A}$ gilt, dass aus $\mathcal{A} \models F$ immer folgt, dass $\mathcal{A} \models G$ (siehe Definition 1.29). In diesem Fall schreiben wieder $F \models G$.

Eine Formel G ist eine **syntaktische Folgerung** von F, wenn wir G us F herleiten können mittels der der Regeln S. In diesem Fall schreiben wir wieder $F \vdash G$ (oder $F \vdash_S G$, wenn wir betonen wollen, dass wir Regeln von S verwenden.) Vergleiche dazu den Text bei Bemerkung 1.32).

Im Folgenden stellen wir uns $\mathcal{F}$ normalerweise als eine Menge von Formeln in der Prädikatenlogik vor. Man bekommt eine Vorstellung von der "Verwendung von S", wenn man an Resolution denkt: Resolutionen sind Folgerungen, siehe Lemma 1.36.

Eine der Errungenschaften Turings bestand darin, dass er uns ein besseres Verständnis dafür vermittelte, was "syntaktische Folgerungen ziehen" bedeuten kann: nämlich eine Turingmaschine zu nutzen. Heute ist das klar: einen Computer nutzen. Aber programmierbare Computer waren damals noch nicht erfunden. Wenn man so möchte, hat Turing seine Maschine erfunden, *um* dem Begriff "syntaktische Folgerung" zu konkretisieren.

Für die Aussagenlogik gibt es keinen Unterschied zwischen den beiden Arten von Folgerungen: Alles, was in der Aussagenlogik wahr ist, kann in der Aussagenlogik bewiesen werden (z.B. wieder durch Resolutionskalkül). Darüber ist alles, was sich so beweisen lässt, auch wahr in der Aussagenlogik.

Jetzt sind wir darauf vorbereitet, Gödels berühmte Ergebnisse präzise zu formulieren.

4.4 Gödels Vollständigkeitssatz

Um die Unvollständigkeitssätze besser zu verstehen, lohnt es sich, ein weiteres gefeiertes Ergebnis von Gödel anzusehen:

Satz 4.8. *Sei $\mathcal{F}$ eine Menge von Formeln in der Prädikatenlogik und G eine Formel.*

$$\mathcal{F} \models G \text{ impliziert } \mathcal{F} \vdash G$$

Sehr naiv könnte man denken, der Vollständigkeitssatz ("Alles kann bewiesen werden") und die Unvollständigkeitssätze ("Nicht alles kann bewiesen werden") würden sich widersprechen. Eine vorsichtigere Formulierung ist diese: "Alle gültigen Sätze können bewiesen werden." (Vollständigkeitssatz) versus "Es gibt Sätze, die weder wahr sind noch falsch" (Unvollständigkeitssatz). Das ist nur eine grobe Interpretation, aber es hilft vielleicht dabei, sich den Unterschied klar zu machen.

Tatsächlich bedeutet 4.7 $\mathcal{F} \models G$ per Definition: wann immer $\mathcal{A} \models \mathcal{F}$ dann $\mathcal{A} \models G$ (G ist eine semantische Konsequenz.) Und $\mathcal{F} \vdash G$ bedeutet: wir können G aus $\mathcal{F}$ im formalen System ableiten. Oder, vereinfachend formuliert:

Alles, was in der Prädikatenlogik wahr ist, kann innerhalb der Prädikatenlogik bewiesen werden

Der nächste Abschnitt zeigt jedoch, dass nicht jede Aussage in der Prädikatenlogik wahr oder falsch ist (im Sinne von "ist semantische Folgerung aus den Axiomen").

4.5 Gödels Unvollständigkeitssätze

Gödels erster Unvollkommenheitssatz besagt, dass kein formales System $(\mathcal{F}, S)$ alle vier der folgenden Eigenschaften haben kann:

1. $(\mathcal{F}, S)$ ist semi-berechenbar,
2. $(\mathcal{F}, S)$ enthält oder impliziert Ganzzahlarithmetik,
3. $(\mathcal{F}, S)$ ist negationsvollständig,
4. $(\mathcal{F}, S)$ ist widerspruchsfrei.

Nummer 1. bedeutet, dass die Menge aller syntaktischen Konsequenzen (von $\mathcal{F}$ unter Verwendung von S) semi-berechenbar (aka rekursiv aufzählbar) ist, siehe Abschnitt 4.1), und dass $\mathcal{F}$ und S selbst es sind. Das heißt, das Problem: "ist eine Formel G eine syntaktische Konsequenz von $\mathcal{F}$?" ist semi-berechenbar (mittels S), vergleiche Definition 4.1.

Zu Nummer 2. siehen unten.

Nummer 3.: **Negationsvollständig** bedeutet, dass für jedes F (ausdrückbar in der Sprache von $\mathcal{F}$) entweder $\mathcal{F} \vdash F$ gilt oder $\mathcal{F} \vdash \neg F$. (Dies ist eine *andere* Bedeutung von "vollständig" als in Satz 4.8). Deshalb nennen wir es hier auch anders. (In der Literatur oder auf wikipedia werden leider oft beide Bedeutungen mit "vollständig" bezeichnet.)

Nummer 4. **Widerspruchsfrei** bedeutet, dass es kein F (ausdrückbar in in der Sprache von $\mathcal{F}$) gibt, so dass sowohl $\mathcal{F} \vdash F$ als auch $\mathcal{F} \vdash \neg F$.

Nummer 2. bedeutet, dass $(\mathcal{F}, S)$ reich genug ist, um die Axiome für das Rachnen mit den natürlichen Zahlen $\mathbb{N}$ zu enthalten, insbesondere Addition und Multiplikation. Eine Möglichkeit, dies in der Prädikatenlogik zu erreichen, sind die **Peano-Axiome**. Normalerweise erfordert dies die Verwendung unendlich vieler prädikatenlogischer Formeln. Eine Möglichkeit verwendet Prädikatenlogik mit = (Identität), eine Konstante (0), eine Funktion mit einem Argument (S, "Nachfolger", d. h. $S(x) = x+1$) und zwei Funktionen mit zwei Argumenten: $f(x,y)$ (zu verstehen als +) und $g(x,y)$ (zu verstehen als $x \cdot y$). Es gibt sechs Formeln, die als Axiome angegeben sind:

1.	$\forall x\, 0 \neq S(x)$	(lesen: $0 \neq x+1$)
2.	$\forall x \forall y\, S(x) = S(y) \Rightarrow x = y$	(read: $x+1 = y+1 \Rightarrow x = y$)
3.	$\forall x\, f(x,0) = x$	(lesen: $x+0 = x$)
4.	$\forall x \forall y\, f(x,S(y)) = S(f(x,y))$	(lesen: $x+(y+1) = (x+y)+1$)
5.	$\forall x\, g(x,0) = 0$	(lesen: $x \cdot 0 = 0$)
6.	$\forall x \forall y\, g(x,S(y)) = f(g(x,y),x)$	(lesen: $x \cdot (y+1) = x \cdot y + x$)

Darüber hinaus fügen wir für jedes Prädikat P mit $k+1$ Argumenten die Formel

$$\forall y_1, \ldots, y_k \left(\left(P(0, y_1, \ldots, y_k) \wedge \forall x \left(P(x, y_1, \ldots, y_k) \Rightarrow P(S(x), y_1, \ldots, y_k) \right) \right) \Rightarrow \forall x\, P(x, y_1, \ldots, y_k) \right)$$

zu den Axiomen hinzu. Daher gibt es insgesamt abzählbar unendlich viele Axiome (aber sie sind semi-berechenbar). Diese letzteren Formeln verwirklichen das **Prinzip der Induktion** ("Vollständige Induktion")

Es gibt mehrere Möglichkeiten, das nächste Ergebnis zu formulieren. Ein üblicher Weg ist dieser.

Satz 4.9 (Gödels erster Unvollständigkeitssatz, 1931)**.** *Jedes widerspruchsfreie und semi-berechenbare formale System $(\mathcal{F}, S)$, das die Ganzzahlarithmetik enthält, ist nicht negationsvollständig.*

In dieser Form bedeutet es, dass es Formeln F gibt, die in der Sprache von $(\mathcal{F}, S)$) ausdrückbar sind, wobei weder $\mathcal{F} \vdash_S F$ noch $\mathcal{F} \vdash_S \neg F$. Mit der Nummerierung 1-4 oben heißt es in dieser Form: "Wenn 1,2 und 4 gilt, dann nicht 3". Natürlich gibt es jetzt gleichwertige Möglichkeiten, dasselbe Ergebnis zu formulieren: "Wenn 1,2 und 3 gilt, dann nicht 4"; oder "Wenn 2,3 und 4 gilt, dann nicht 1". Es ist hilfreich, für jeden Fall Beispiele zu berücksichtigen.

Nicht negationsvollständig. Dies war der "klassische" Fall, der die Experten wie David Hilbert oder John von Neumann überraschte: unter Verwendung der Peano-Axiome können wir nicht erwarten, eine gegebene Aussage in einem formalen System zu beweisen oder zu widerlegen. Es gibt also einige Aussagen F, die weder "wahr" noch "falsch" sind, zumindest nicht aus dem System heraus betrachtet. (Denn sonst könnten wir sie ja wegen Satz 4.8 beweisen.) Tatsächlich können wir in einem solchen Fall F zu den Axiomen hinzufügen, ohne die Widerspruchsfreiheit zu verlieren. Und wir können stattdessen auch $\neg F$ zu den Axiomen hinzufügen, ohne die Widerspruchsfreiheit zu verlieren. (Aber natürlich nicht beides.)

Ein berühmtes Beispiel für eine solche Aussage ist die Kontinuumshypothese in den Zermelo-Frenkel-Axiomen (zu letzterem siehe Abschnitt 5.)

Nicht widerspruchsfrei. Nehmen Sie die obigen Peano-Axiome als unser $\mathcal{F}$ und definieren Sie eine (sehr liberale) Regel S, die besagt: "Für jede Formel F haben wir $\mathcal{F} \vdash F$". Daher ist jede Formel eine Folge von $\mathcal{F}$ unter S. Das bedeutet insbesondere dass $\mathcal{F} \vdash_S F$ und $\mathcal{F} \vdash_S \neg F$ für jedes F. Dieses Beispiel ist alles andere als widerspruchsfrei.

Nicht effektiv aufzählbar. Es ist etwas ganz anderes, Folgerungen in einem formalen System zu ziehen (das viele verschiedene Modelle haben kann), als Folgerungenn in einem bestimmten konkreten Modell $\mathcal{A}$ zu ziehen. Die Definition von $\mathcal{A}$ ermöglicht es uns zu entscheiden, ob F in $\mathcal{A}$ wahr ist oder nicht. Nehmen wir also die **echte Arithmetik** an, also die Struktur $\mathcal{A}$ für die Peano-Axiome, bei denen $U^{\mathcal{A}} = \mathbb{N}$ ist, das 0-Symbol die tatsächliche 0 bedeutet, $S^{\mathcal{A}}(x) = x+1$ ist, $f^{\mathcal{A}}(x,y) = x+y$ und so weiter. (Jedes Symbol bedeutet das, was es bedeuten soll.) Nun gilt für jede Aussage, die wir in der Sprache von $\mathcal{F}$ ausdrücken können, dass sie *in diesem Modell* entweder wahr oder falsch ist.[2]) Darüber hinaus kann keine Aussage gleichzeitig wahr und falsch sein. Wir haben also Ganzzahlarithmetik (2), Vollständigkeit (3) und Konsistenz (4). Nach dem Satz 4.9 gibt es dazu kein formales System, das semi-berechenbar ist.

Enthält keine Ganzzahlarithmetik. Es ist eine interessante (und tiefgreifende) Tatsache, dass es auch vollständige (und widerspruchsfreie) Theorien gibt. Ein einfaches Beispiel ist die Aussagenlogik. Anspruchsvollere Beispiele sind etwa die **Presburger-Arithmetik** (ganzzahlige Arithmetik nur mit +, also ohne Multiplikation), oder die Axiome der euklidischen Geometrie, oder monadische Prädikatenlogik (alle Prädikate haben nur ein Argument, vergleiche Übungen).

[2]Für einige Aussagen konnte bisher keine Antwort gefunden werden, etwa "Ist jede gerade Zahl größer als 2 Summe zweier Primzahlen?" oder "Gibt es unendlich viele Primzahlzwillinge?". (Primzahlzwillinge sind Primzahlen der Form p, $p+2$, wie 11 und 13.) Dennoch sind diese Aussagen in $(\mathbb{N}, +, \cdot)$ entweder wahr oder falsch.

Satz 4.10 (Gödels zweiter Unvollständigkeitssatz, 1931)**.** *Die Widerspruchsfreiheit eines widerspruchsfreien und semi-berechenbaren formalen Systems $(\mathcal{F},S)$, das Ganzzahlarithmetik enthält, kann als eine Formel F in der Sprache von $\mathcal{F}$ geschrieben werden.*
Es gilt dann $\mathcal{F} \nvdash_S F$.

5 Zermelo-Fraenkel axioms and the axiom of choice

30 Jan

In the beginning of the 20th century a lot of researchers tried to develop an axiomatization of the foundations of mathematics. A milestone was the book *Principia Mathematicae* by *Bertrand Russell* and *Alfred North Whitehead*, see for instance the interesting comic book *Logicomix*. It became clear that the most basic notion is that of a set. Once a set is properly defined one can easily define numbers, relations, functions etc.

For instance, once sets are defined, the natural numbers can be defined as follows:

$$0 = \{\}, \quad S(n) = n \cup \{n\},$$

where S is the successor function (think: "$n+1$"). This means the first natural numbers are $0 = \{\} = \varnothing, 1 = \{0\} = \{\varnothing\}, 2 = \{0,1\} = \{\varnothing, \{\varnothing\}\}, 3 = \{0,1,2\} = \{\varnothing, \{\varnothing\}, \{\varnothing, \{\varnothing\}\}\}$.... Once the successor function S is defined we may define addition by

$$f(n,m) = n+m = S^n(m) = S(S(\cdots S(m)\cdots)$$

and multiplication by

$$n \cdot 0 = g(n,0) = 0, \quad n \cdot S(m) = g(n,S(m)) = S^n(g(n,m)).$$

The latter means for instance (note that $2 = S(1)$)

$$3 \cdot 2 = S^3(3 \cdot 1) = S^3(S^3(1 \cdot 0)) = S^3(S^3(0)) = 6$$

One can easily check that these definitions satisfy the Peano axioms.

Relations on a set W are just subsets of $W \times W$, see the previous sections. Functions $f : X \to Y$ are an assignment of elements $f(x) \in Y$ for each $x \in X$. And so on. It remains to define what a set is.

Earlier approaches to define sets were given in a non-formal way, e.g. Georg Cantor's *naïve set theory*. It went along the lines of "a set is a collection of objects".

Unter einer Menge verstehen wir jede Zusammenfassung von bestimmten wohl unterschiedenen Objekten unserer Anschauung oder unseres Denkens zu einem Ganzen.

Hence a set is everything that can be defined using language. Even though several aspects of Cantor's work are milestones of logic and mathematics (Cantor's diagonal argument, uncountable sets...) the insufficiency of a naive definition of sets was revealed by **Russell's paradox**:

Let R be the set of all sets that are not members of themselves. If R is not a member of itself, then its definition implies that it must contain itself. If R contains itself, then it contradicts its own definition. Symbolically:

$$\text{Let } R = \{x \mid x \notin x\}, \text{ then } R \in R \Leftrightarrow R \notin R \equiv \neg(R \in R)$$

Russell's paradox (also found earlier by Zermelo) and Hilbert's demand to find an axiomatization for all of mathematics lead people to develop an axiomatization of set theory.

Maybe the most standard definition today is called the **Zermelo-Fraenkel-axioms** (ZF), developed by Ernst Zermelo in 1908 and later improved by Fraenkel and Skolem in order to allow to prove certain concepts ("cardinal numbers") and avoid certain improperly defined terms. Nowadays these axioms can be (and usually are) stated in first-order logic. There are several equivalent Formulations. One is the following:

5.1 Zermelo-Fraenkel axioms

ZF uses identity $=$ and one further binary predicate $P(x,y)$, namely "x is element of y", short (as usual) $x \in y$. Hence we will write $x = y$ and $x \in y$ in the sequel rather than $P(x,y)$. Note that this allows us to define a further predicate "subsets": a set z is a subset of a set x if and only if every element of z is also an element of x: $z \subseteq x$ means: $\forall q\ (q \in z \Rightarrow q \in x)$.

In the sequel, keep in mind that the intended universe is "all sets": the axioms are tailored in a way such that each model behaves like its universe being sets.

Axioms number 1 to 5 are the "intuitive" axioms that we would expect from the properties of sets that we are used to. Numbers 6 to 9 are rather less intuitive, but needed in order to avoid certain problems (all axioms are accompanied by a short explanation of their respective meaning). This section makes strong use of the corresponding wikipedia article, but it is not exactly the same.

1. Axiom of extensionality Two sets are equal (are the same set) if they have the same elements.

$$\forall x\, \forall y\, \big(\forall z(z \in x \Leftrightarrow z \in y) \Rightarrow x = y\big).$$

2. Empty set axiom There is a set without elements.

$$\exists x\, \neg \exists y\, y \in x$$

Notation: this x is called $\varnothing$.

3. Axiom of pairing If x and y are sets, then there exists a set which contains x and y as elements.

$$\forall x\, \forall y\, \exists z\, (x \in z \wedge y \in z).$$

For instance, the pairing of $x = \{1,2\}$ and $y = \{2,3\}$ is $z = \{\{1,2\},\{2,3\}\}$.

4. Axiom of union The union over the elements of a set exists. More precisely, for any set of sets x there is a set y containing every element of every element of x.

$$\forall x\, \exists y\, \forall u\, \big(u \in y \Leftrightarrow (\exists v\, (v \in x \wedge u \in v))\big)$$

For example, the union over the elements of the elements of $x = \{\{1,2\},\{2,3\}\}$ is $y = \{1,2,3\}$.

5. Axiom of power set This axiom states that for any set x, there is a set y that contains every subset of x (compare Def. 3.17):

$$\forall x\, \exists y\, \forall z\, \big(z \subseteq x \Rightarrow z \in y\big).$$

Ensures the existence of the set of all subsets of x, the *power set* (notation: $\mathcal{P}(x)$) of any set x. For instance if $x = \{1,2,3\}$ then $\mathcal{P}(x) = \{\varnothing, \{1\}, \{2\}, \{3\}, \{1,2\}, \{1,3\}, \{2,3\}, \{1,2,3\}\}$.

6. Axiom of specification Subsets are often denoted by something like $\{n \in \mathbb{N} \mid n \text{ prime }\}$, or $\{n \in \mathbb{N} \mid n \equiv 2 \bmod 5\}$, or so. This axiom ensures that such subsets always exist.

Let F be any Formel in the language of ZF with some free variable x (y is not free in F). Then:

$$\forall z\, \exists y\, \forall x\, \big(x \in y \Leftrightarrow (x \in z \wedge F)\big).$$

Since this axiom is of the form "for any F" it is in fact an infinite collection of axioms in first-order logic (compare the Peano axiom of induction).

This axiom ensures that we can write $y = \{n \in \mathbb{N} \mid n \text{ prime}\}$ using $z = \mathbb{N}$ and $F = P'(n) =$"n is prime".

This axiom prevents the "set" R of the Russell paradox from being a set!

Let us assume the set M of all sets is a set. Let $P'(x) = x \notin x = \neg(x \in x)$. Then by this axiom $R = \{x \in M \mid P'(x)\}$ is a set. But we know that it cannot be a set, since its existence leads to a contradiction. Hence our assumption is wrong and the set of all sets does not belong to our universe, that is, M is not a set (and neither is R).

7. Axiom of replacement The axiom of replacement asserts that the image of a set under any definable function will also fall inside a set.

Formally, let F be a Formel in the language of ZF whose free variables are x and y. Then:

$$\forall x\, \exists y\, \forall u \big(u \in y \Leftrightarrow \exists z\, (z \in x \wedge F(z,u))\big).$$

Spelled out this means: For each set x exists a set y consisting of all elements u for which there is $z \in x$ such that $F(z,u)$ holds.

In particular this means that the image of a set under some function f is again a set: choose $F(z,u)$ as $u = f(z)$. Then y contains all u such that $f(z) = u$, where z takes all values in x.

Again this axiom is of the form "for any F", hence it is in fact an infinite collection of axioms in first-order logic (compare the Peano axiom of induction).

8. Axiom of infinity There are infinite sets. More precisely:

$$\exists x\, \big(\varnothing \in x \wedge \forall y\, (y \in x \Rightarrow y \cup \{y\} \in x)\big).$$

This axiom ensures the existence of infinite sets like the natural numbers in the set theoretic definition above: $\{\varnothing, \{\varnothing\}, \{\varnothing, \{\varnothing\}\}, \cdots, \}$. Together with the power set axiom it ensures also the existence of uncountably infinite sets.

9. Axiom of foundation Also known as "axiom of regularity". Every non-empty set x contains a member y such that x and y are disjoint sets.

$$\forall x \left(x \neq \varnothing \Rightarrow \exists y \left(y \in x \wedge \neg \exists z \left(z \in y \wedge z \in x \right) \right) \right)$$

This implies, for example, that no set is an element of itself. More generally, it prevents the existence of cyclic chains of subsets: $x_1 \subset x_2 \subset \cdots \subset x_n \subset x_1$.

5.2 Axiom of choice

Now every model for the ZF-axioms has the properties that we require for the notion of "sets", without leading to some non-desired effects (as mentioned above: e.g. a set cannot be its own element, the set of all sets is not a set, ...). Still it was found that one rule is missing:

Axiom of choice For any set x of nonempty sets x_i there is a function that chooses one element from each x_i.

$$\forall x \big(\varnothing \notin x \Rightarrow \exists f \colon x \to \bigcup x \quad \forall u \in x \, f(u) \in u \big).$$

It seems rather intuitive that such a rule must indeed be true: For instance, given the set

$$\{\{1,4,5\},\{12,31,44,77\},\{101,202,303\}\}$$

we can choose one element from each set, for instance 1, 12, 101. Sometimes the axiom of choice is illustrated as follows:

Assume you are the king of n provinces with finitely many citizens each. You need to choose a governour for each province. How do you proceed? Simple: just choose the oldest citizen to be governour.

What if you have infinitely many provinces with finitely many citizens each? The same procedure works. What if you have infinitely many provinces with infinitely many citizens each? There may not longer be an oldest citizen... can you still Formelte a general rule for this case?

Now consider an even worse case: you are supposed to choose one element from each subset of the real numbers. There is no known rule how to achieve this. But the axiom of choice states that such a rule exists. Even though it seems so intuitive this rule does not follow form the ZF axioms 1-9:

Satz 5.1 (Gödel 1940, Cohen 1963)**.** *The axiom of choice is not a consequence of the Zermelo-Fraenkel axioms. Neither is its negation a consequence of the Zermelo-Fraenkel axioms.*

Hence nowadays **ZFC**, that is, ZF together with the axiom of choice, are regarded as the (best known?) standard axiomatization of set theory.

Given the ZF axioms there are several equivalent Formeltions of the axiom of choice. Here are two of them:

Well-ordering Theorem For any set X there is a linear order, that is: there is a relation $\leq$ on X that is

- antisymmetric; that is if $a \leq b$ and $b \leq a$ then $a = b$.
- transitive; that is if $a \leq b$ and $b \leq c$ then $a \leq c$.

- total; that is $a \leq b$ or $b \leq a$ for all a, b.

Intuitively one may doubt that this can be true: think of the set $\mathbb{C}$, or of $\mathbb{R}^2$. How can there possibly be a well defined relation like $\leq$?

Zorn's Lemma Let $(M, \leq)$ be a partially ordered set (i.e. $(M, \leq)$ is reflexive, antisymmetric and transitive). If every subset of M that has a linear order has an upper bound in M then the set M contains at least one maximal element.

The fact that the axiom of choice (resp. its equivalent Formeltions) are controversial (see wikipedia) is reflected in the following quotation:

The Axiom of Choice is obviously true, the well-ordering principle obviously false, and who can tell about Zorn's lemma?

The controversy maybe inspired people to prove bizarre and counterinutuitive results using the axiom of choice. See for instance the Banach-Tarski paradox.

Complexity

In Theoretical Computer Science there are several levels of complexity of a (class of) problem(s): P, NP,... The question whether a Formel in first-order logic is erfüllbar is harder than any of them. Let us briefly sketch this hierarchy. A problem is here a problem that can have infinitely many, and arbitrary large, inputs, like an integer, or a graph, or a tree, or a list of integers. The size of the input is measured by some reasonable measure, like the number of the digits of an integer, or the number of vertices of a graph or tree, or the number of the entries of some list.

P A problem is in P if there is an algorithm that solves the problem for any input of size n in polynomial time; i.e., the number of the required steps is in $O(p(n))$ where p is some polynomial (or in general, something that is smaller than some polynomial). E.g. sorting a list of n numbers can be done in $O(n^2)$ steps (quicksort), or even in $O(n \log n)$ (mergesort). Finding the minimal number in a list of integers can be done trivially in time $O(n)$. The problem PRIME, i.e., testing whether a number x with n binary digits is a prime number requires naively $O(2^{n/2})$ steps (test all odd numbers between 3 and $\sqrt{x}$ if they divide x). If we couldn't do better problem this would not be in P, but there are algorithms known that need $O(n^{12})$ steps. Hence PRIME is in P.

NP A problem is in NP if each correct solution (aka "witness", or "certificate") of it can be checked in polynomial time (wrt the size n of the input). For instance, the question "does a given graph G with n vertices have a Hamiltonian cycle" is in NP. (A graph has a Hamiltonian cycle if it contains a cycle that contains each vertex of G exactly once.) In this example a solution is such a Hamiltonian cycle, and it is easily checked whether a given solution is correct. On the other hand, finding a Hamiltonian cycle can be pretty hard in general.

Similarly, the prime factorization of integers with n digits is in NP: it might be hard to tell what the prime factors of 1003 are, or of 1007 (are they prime numbers or composite numbers?) But a witness for the factorization of 1007 is $(19, 53)$, since $19 \cdot 53 = 1007$. (And $1003 = 17 \cdot 59$.)

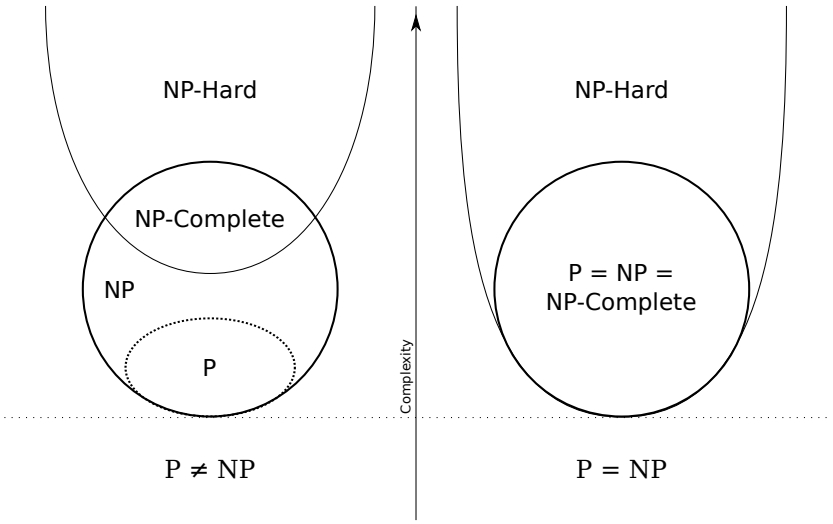


Figure 2: The situation if P≠NP (left) and if P=NP (right).

It is a big open problem whether P=NP. It is generally assumed that NP is a bigger class than P. A problem is called **NP-hard** if a solution of this problem can be translated into a solution for any problem in NP in polynomial time. A problem that is both in NP and NP-hard is called **NP-complete**. Figure 2 illustrates the situation (for both cases, P=NP and P≠NP). It is known that the Hamiltonian cycle problem is NP-complete, but prime factorization of integers is not. Usually such results are shown by reducing a problem to SAT, the question whether a Formel in propositional logic in KNF is erfüllbar, compare Remark 1.20. SAT was the first problem known to be NP-complete.

Beyond NP Are there problems that are even harder than problems in NP? Yes. In Section sec:computable we saw unberechenbar problems. As a side remark, a class of problems that is harder than NP problems is the class NEXP. This is the class of problems having instances where all witnesses are of exponential size. It is known that NP≠NEXP (more precisely, NP is a proper subset of NEXP). The examples for this look somehow artificial. For instance, the question whether a non-deterministic Turing machine will stop after n steps is in NEXP, since each witness is already of size $O(2^n)$. In a similar manner, it is known that certain graphs with 2^n vertices can be encoded into logical circuits with $O(n)$ logic gates ("succinct circuits"). If one asks for a Hamiltonian cycle in such a graph this problem is in NEXP, since each witness is of size $O(2^n)$. In the sequel we will see that there are even harder problems, namely, unberechenbar problems. One is the question whether a given Formel in first-order logic is erfüllbar.

Literature

- Uwe Schöning: Logic for Computer Scientists *(covers most of Sections 1 and 2 in a very compact but comprehensive manner)*

- Uwe Schöning: Logik für Informatiker (same in German)

- Martin Kreuzer, Stefan Kühling: Logik für Informatiker (German) *(one of the few textbooks covering Section 4)*

- H.-D. Ebbinghaus, J. Flum, W. Thomas: Mathematical Logic *(THE classic textbbok on formal logic, contains a lot more than this lecture)*
- Wolfgang Rautenberg: A Concise Introduction to Mathematical Logic *(another comprehensive textbook)*

- M. Sipser: Introduction to the theory of computation *(contains the complete proof of Theorem 4.4)*
- A.K. Doxiades, C.H. Papadimitriou, A. Papadatos: Logicomix *(Just for fun: a comic book telling the story of Russell (and how he met Gödel and Whitehead and Cantor...)*