Diskrete Mathematik

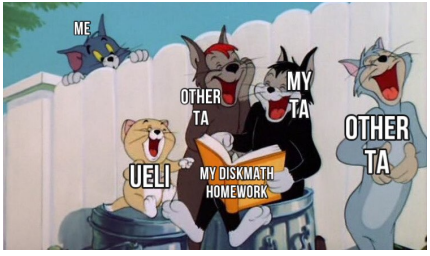
Woche 12

Shivram Sambhus (cs.shivi.io)

ETH Zürich

Ende Kapitel 6

- **Thema:** Kapitel 6 - Logik (Finale).
- **Fokus:** Ein mächtiges Theorem und seine Anwendungen.
- **Letzte Woche:**
  - Proof Systems, Syntax vs. Semantik
  - Aussagenlogik: CNF/DNF
  - Prädikatenlogik: Quantoren, Strukturen
- **Diese Woche:**
  - Prenex Normal Form & Skolemisierung
  - **Theorem 6.12:** Russell, Cantor, Gödel, Turing in einem!
  - Resolution Calculus (Aussagen- & Prädikatenlogik)
  - Theorem Proving Patterns



Roadmap

1. **Teil 1: Prenex Normal Form (PNF)** Alle Quantoren nach vorne – Standardform für Prädikatenlogik.
2. **Teil 2: Skolemisierung** Existenzquantoren eliminieren für automatisches Beweisen.
3. **Teil 3: Das Mächtige Theorem (Theorem 6.12)** Russell's Paradox, Cantor, Unberechenbarkeit – alles in einem!
4. **Teil 4: Resolution Calculus** Der Kalkül für Aussagen- und Prädikatenlogik.
5. **Teil 5: Beweispatterns (Cheatsheet)** Kurzübersicht der 5 wichtigsten Exam-Strategien.

Teil 1: Prenex Normal Form

Motivation: Warum Standardformen?

Letzte Woche haben wir gesehen:

- **Syntax vs. Semantik:** $\vdash$ vs. $\models$
- **Ziel:** Beweise mechanisch führen (Kalküle)

Problem: Prädikatenlogik-Formeln können beliebig verschachtelt sein:

$$\neg\forall x P(x) \land \exists x Q(x)$$

Idee: Bringe Formeln in eine **Standardform**, um sie systematisch zu analysieren.

$\Rightarrow$ Alle Quantoren an den Anfang = **Prenex Normal Form**

Definition 6.38 (Prenex Normal Form): Eine Formel ist in **Prenex Normal Form (PNF)**, wenn sie die Form hat:

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

wobei $Q_i \in \{\forall, \exists\}$ und F quantorenfrei ist.

Theorem 6.10: Existenz der PNF

Theorem 6.10: Jede Formel der Prädikatenlogik ist äquivalent zu einer Formel in Prenex Normal Form.

Idee:

1. Variablen umbenennen, um Konflikte zu vermeiden.
2. Negationen nach innen schieben (De Morgan für Quantoren).
3. Quantoren nach aussen ziehen (Quantor-Verschiebungsregeln).

Regeln für die PNF-Transformation

De Morgan für Quantoren:

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

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

Quantoren über Junktoren ziehen: Falls x **nicht frei** in G:

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

$$(\forall x F) \vee G \equiv \forall x (F \vee G)$$

$$(\exists x F) \wedge G \equiv \exists x (F \wedge G)$$

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

Achtung: Falls x frei in G ist, muss man x erst umbenennen!

Beispiel: Transformation zu PNF

Formel: $F = \neg\forall x P(x) \vee (\exists y Q(y) \wedge R(z))$

Schritt 1: De Morgan auf $\neg\forall x P(x)$:

$$F \equiv \exists x \neg P(x) \vee (\exists y Q(y) \wedge R(z))$$

Schritt 2: Inneren Quantor rausziehen. z ist frei, aber y nicht in $\neg P(x)$, also ok:

$$F \equiv \exists x(\neg P(x) \vee (\exists y Q(y) \wedge R(z)))$$

$$F \equiv \exists x \exists y(\neg P(x) \vee (Q(y) \wedge R(z)))$$

Ergebnis (PNF):

$$F \equiv \exists x \exists y(\neg P(x) \vee (Q(y) \wedge R(z)))$$

Beispiel: Umbenennung nötig

Formel: $F = (\forall x P(x)) \wedge Q(x)$

Problem: x ist frei in $Q(x)$ und gebunden in $\forall x P(x)$. Wir können den Quantor **nicht** einfach rausziehen!

Lösung: Umbenennen.

$$F \equiv (\forall y P(y)) \wedge Q(x)$$

Jetzt können wir den Quantor rausziehen:

$$F \equiv \forall y (P(y) \wedge Q(x))$$

Regel: Immer zuerst Variablen umbenennen, um Konflikte zu vermeiden.

Übungsblock 1: Prenex Normal Form

Aufgabe 1 (HS22): Finde eine äquivalente Formel in Prenex-Form:

$$(\forall x P(x)) \to (\exists x Q(x) \vee P(y))$$

Aufgabe 2 (HS19): Bringe in PNF:

$$\neg \forall x\, P(x) \vee \neg Q(y) \wedge \exists y\, (P(x) \vee Q(y))$$

Aufgabe 3 (HS18): Bringe in PNF:

$$P(x, x) \wedge \exists x\, ((\forall x\, P(x, y)) \to Q(x))$$

Lösungen Übungsblock 1

Zu Aufgabe 1:

- Implikation: $\neg(\forall x P(x)) \vee (\exists x Q(x) \vee P(y))$
- De Morgan: $(\exists x \neg P(x)) \vee (\exists x Q(x) \vee P(y))$
- Umbenennen: $(\exists x \neg P(x)) \vee (\exists z Q(z) \vee P(y))$
- Quantoren raus: $\exists x \exists z (\neg P(x) \vee Q(z) \vee P(y))$

Zu Aufgabe 2:

- Klammern beachten! $\neg \forall x P(x) \vee (\neg Q(y) \wedge \exists y (P(x) \vee Q(y)))$
- De Morgan: $\exists x \neg P(x) \vee (\neg Q(y) \wedge \exists w (P(x) \vee Q(w)))$
- Quantoren: $\exists x \exists w (\neg P(x) \vee (\neg Q(y) \wedge (P(x) \vee Q(w))))$

Lösungen Übungsblock 1 (Fortsetzung)

Zu Aufgabe 3:

- Die Formel hat mehrere Variablenkonflikte!
- 1. Äusseres x in $P(x, x)$ ist frei, inneres x im $\exists$ ist gebunden.
- 2. Innerstes $\forall x$ hat nochmal x.
- Schritt 1: Umbenennen.
- $P(x, x) \wedge \exists u\,((\forall v\, P(v, y)) \rightarrow Q(u))$
- Schritt 2: Implikation eliminieren.
- $P(x, x) \wedge \exists u\,(\neg\forall v\, P(v, y) \vee Q(u))$
- $= P(x, x) \wedge \exists u\,(\exists v \neg P(v, y) \vee Q(u))$
- Schritt 3: Quantoren rausziehen.
- $\exists u \exists v\,(P(x, x) \wedge (\neg P(v, y) \vee Q(u)))$

Teil 2: Skolemisierung

Motivation: Existenzquantoren eliminieren

Wir haben jetzt Formeln in PNF:

$$\forall x \exists y \forall z \, F(x, y, z)$$

Problem: Der $\exists$-Quantor ist schwierig zu handhaben. "Es existiert ein y" – aber welches?

Idee: Mache die Abhängigkeit **explizit**! Wenn y von x abhängt, führe eine Funktion $f(x)$ ein, die y "auswählt".

$\Rightarrow$ **Skolemisierung** eliminiert $\exists$-Quantoren.

Idee:

- $\forall$-Quantoren können wir "droppen" (implizit universal).
- $\exists$-Quantoren ersetzen wir durch **Skolem-Funktionen**.

Skolemisierung

Definition (Skolem-Funktion): Wenn $\exists y$ im Scope von $\forall x_1, \ldots, \forall x_n$ steht, ersetzen wir y durch $f(x_1, \ldots, x_n)$, wobei f ein neues Funktionssymbol ist.

Beispiel:

$$\forall x \exists y \, P(x, y)$$

Skolemisiert:

$$\forall x \, P(x, f(x))$$

Intuition: "y hängt von x ab" wird explizit gemacht.

Wichtig: Skolemisierung erhält **Erfüllbarkeitsäquivalenz**, nicht logische Äquivalenz!

Beispiel: Vollständige Transformation

Formel: $F = \forall x \exists y \forall z (P(x, y) \vee Q(y, z))$

Schritt 1: Skolemisierung. y hängt von x ab: $y \to f(x)$.

$$F' = \forall x \forall z (P(x, f(x)) \vee Q(f(x), z))$$

Schritt 2: Universelle Quantoren droppen.

$$P(x, f(x)) \vee Q(f(x), z)$$

Schritt 3: Universelle Quantoren "droppen" (implizit).

$$P(x, f(x)) \vee Q(f(x), z)$$

Warum Skolemisierung? Equisatisfiability

Wichtige Eigenschaft: Original und Skolem-Form sind **equisatisfiable** (gleiche Erfüllbarkeit).

$$\exists x\, P(x) \text{ erfüllbar} \iff P(c) \text{ erfüllbar}$$

Warum reicht das? Erinnere: $F \vDash G \iff \{F, \neg G\}$ ist **unerfüllbar**.

Für Widerspruchsbeweise brauchen wir nur Erfüllbarkeit zu prüfen!

Ausblick: Automatisches Beweisen

Mit PNF und Skolemisierung können wir Prädikatenlogik-Formeln in eine Form bringen, die für automatische Verfahren geeignet ist.

Der Weg:

$$\text{Formel} \xrightarrow{\text{PNF}} \text{Prenex} \xrightarrow{\text{Skolem}} \text{Nur-}\forall \xrightarrow{\text{Klauseln}} \text{CNF}$$

Bemerkung: Resolution kann auch für Prädikatenlogik erweitert werden (mit *Unifikation*), aber das behandeln wir hier nicht im Detail.

Übungsblock 2: Skolemisierung

Aufgabe 1: Skolemisiere:

$$\forall x \exists y \forall z \, (P(x, y) \vee \neg Q(y, z))$$

Aufgabe 2: Skolemisiere:

$$\exists x \forall y \exists z \, R(x, y, z)$$

Aufgabe 3: Warum erhält Skolemisierung nur Erfüllbarkeitsäquivalenz, nicht logische Äquivalenz? Gib ein konkretes Gegenbeispiel.

Lösungen Übungsblock 2

Zu Aufgabe 1:

- Skolemisiere: y hängt von x ab $\Rightarrow y \to f(x)$.
- $\forall x \forall z \, (P(x, f(x)) \vee \neg Q(f(x), z))$

Zu Aufgabe 2:

- x hat keine $\forall$ davor $\Rightarrow$ Skolem-Konstante c.
- z hängt von y ab $\Rightarrow z \to g(y)$.
- Ergebnis: $\forall y \, R(c, y, g(y))$

Lösungen Übungsblock 2 (Fortsetzung)

Zu Aufgabe 3:

- Betrachte $\exists x\, P(x)$ und seine Skolemisierung $P(c)$.
- $\exists x\, P(x)$ sagt: "Es gibt ein x mit $P(x)$."
- $P(c)$ sagt: "c erfüllt P." (Fixiert ein spezifisches Element)
- Die Formeln sind **nicht logisch äquivalent**:
- $P(c) \models \exists x\, P(x)$, aber $\exists x\, P(x) \not\models P(c)$.
- Aber: $\exists x\, P(x)$ erfüllbar $\iff$ $P(c)$ erfüllbar.
- Das reicht für Widerspruchsbeweise!

Teil 3: Das Mächtige Theorem

Zwischenstand: Was haben wir bisher?

Wir haben Werkzeuge für Prädikatenlogik kennengelernt:

- **PNF:** Quantoren an den Anfang
- **Skolemisierung:** $\exists$ eliminieren

Jetzt kommt etwas Überraschendes: Ein einzelnes Theorem der Prädikatenlogik erklärt fundamentale Grenzen!

Motivation: Eine überraschende Verbindung

Was haben diese vier Resultate gemeinsam?

1. **Russell's Paradox (1901):** Die Menge aller Mengen, die sich nicht selbst enthalten, kann nicht existieren.
2. **Cantors Diagonalargument (1891):** Die Menge $\{0,1\}^\infty$ ist überabzählbar.
3. **Gödels Unvollständigkeitssatz (1931):** Jedes konsistente formale System ist unvollständig.
4. **Turings Halteproblem (1936):** Es gibt keine Funktion, die entscheidet, ob ein Programm terminiert.

Antwort: Sie alle folgen aus einem einzigen logischen Theorem!

Theorem 6.12: Die Universelle Aussage

Theorem 6.12 (Skript): Die folgende Formel ist eine **Tautologie** der Prädikatenlogik:

$$\neg\exists x\, \forall y \Big(P(y,x) \iff \neg P(y,y) \Big)$$

Äquivalent:

$$\forall x\, \exists y \Big(P(y,x) \iff P(y,y) \Big)$$

In Worten: *"Es gibt kein x, das genau die y 'sammelt', die sich selbst nicht 'sammeln'."*

Dies ist eine rein logische Aussage - sie gilt in **jeder** Struktur, für **jedes** Prädikat P.

Beweis von Theorem 6.12

Wir zeigen: $\neg\exists x\, \forall y(P(y,x) \iff \neg P(y,y))$ ist eine Tautologie.

Äquivalent: $\forall x\, \exists y(P(y,x) \iff P(y,y))$ ist eine Tautologie.

Beweis (durch Äquivalenzumformung): Wir müssen für jedes x ein "Zeuge" y finden, sodass $P(y,x) \iff P(y,y)$.

Wähle $y := x$!

Dann: $P(x,x) \iff P(x,x)$, was trivialerweise wahr ist.

Da für jedes x der Zeuge $y = x$ existiert, ist die Formel in jeder Struktur wahr.

□

Die Magie: Korollar 6.13 (Russell's Paradox)

Korollar 6.13: Es gibt keine Menge R, die genau alle Mengen enthält, die sich nicht selbst enthalten.

Beweis:

- Universum $U =$ Menge aller Mengen.
- Interpretiere $P(y, x)$ als "$y \in x$".
- Theorem 6.12 sagt: $\neg\exists x\, \forall y (y \in x \iff y \notin y)$.

Die Menge $R = \{y : y \notin y\}$ kann also **nicht existieren**.

Das ist Russell's Paradox! Eine rein logische Konsequenz von Theorem 6.12. $\square$

Korollar 6.14: Cantors Diagonalargument

Korollar 6.14: Die Menge $\{0,1\}^\infty$ (alle unendlichen Bitfolgen) ist überabzählbar.

Beweis (mit Theorem 6.12):

- Angenommen $\{0,1\}^\infty$ wäre abzählbar, also $\{0,1\}^\infty = \{s_0, s_1, s_2, ...\}$.
- Universum $U = \mathbb{N}$, Bitfolgen als Funktionen $s : \mathbb{N} \to \{0,1\}$.
- Interpretiere $P(y,x)$ als "$s_x(y) = 1$" (das y-te Bit der x-ten Folge).

Theorem 6.12: $\neg\exists x\, \forall y (s_x(y) = 1 \iff s_y(y) \neq 1)$.

Aber: Definiere die "Anti-Diagonale" d durch $d(y) = 1 \iff s_y(y) \neq 1$. Diese Folge d ist in $\{0,1\}^\infty$, aber $d \neq s_x$ für alle x. Widerspruch! $\square$

Visualisierung: Cantors Diagonalargument

Betrachte die "Tabelle" aller Bitfolgen:

	0	1	2	3	$\cdots$
s_0	$\mathbf{1}$	0	1	0	$\cdots$
s_1	0	$\mathbf{0}$	1	1	$\cdots$
s_2	1	1	$\mathbf{1}$	0	$\cdots$
s_3	0	0	0	$\mathbf{1}$	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$

Die **Diagonale** hat die Werte $1, 0, 1, 1, ...$

Die **Anti-Diagonale** d: Flippe jedes Bit! $d = 0, 1, 0, 0, ...$

d unterscheidet sich von **jeder** Zeile in mindestens einer Position!

Korollar 6.15: Unberechenbare Funktionen existieren

Korollar 6.15: Es gibt Funktionen $f : \mathbb{N} \to \{0, 1\}$, die von keinem Programm berechnet werden.

Intuition: Programme sind endliche Texte (Strings). Wir können alle Strings auflisten: `"a"`, `"b"`, ..., `"aa"`, `"ab"`, ... — das sind **abzählbar** viele!

Aber Funktionen $f : \mathbb{N} \to \{0, 1\}$ sind dasselbe wie unendliche Bitfolgen — und die sind **überabzählbar** (Korollar 6.14)!

Korollar 6.15: Beweis

Beweis:

- Sei $\{P_0, P_1, P_2, ...\}$ eine Aufzählung aller Programme.
- Jedes Programm P_i berechnet eine Funktion $f_i : \mathbb{N} \to \{0, 1\}$.
- Es gibt nur **abzählbar viele** Programme $\Rightarrow$ abzählbar viele berechenbare Funktionen.
- Aber: $|\{0, 1\}^{\mathbb{N}}|$ ist **überabzählbar** (Korollar 6.14).

Kardinalitäts-Argument: $|\text{Programme}| = \aleph_0 < 2^{\aleph_0} = |\text{Funktionen}|$

Also existieren "viel mehr" Funktionen als Programme! $\square$

Korollar 6.16: Das Halteproblem

Korollar 6.16: Es gibt kein Programm H, das für jedes Programm P und Input x entscheidet, ob $P(x)$ terminiert.

Die Frage: Können wir einen "Universal-Debugger" bauen, der vorhersagt ob ein Programm fertig wird oder ewig loopt?

Spoiler: Nein! Und der Grund ist wieder Selbstreferenz...

Korollar 6.16: Beweis-Idee

Angenommen so ein Programm H existiert:

$$H(P, x) = \begin{cases} 1 & \text{falls } P(x) \text{ terminiert} \\ 0 & \text{falls } P(x) \text{ ewig loopt} \end{cases}$$

Wir bauen ein Paradox-Programm D:

```
def D(n):
    if H(P_n, n) == 1:   # Falls P_n(n) hält...
        while True: pass  # ...dann loope ewig!
    else:                 # Falls P_n(n) loopt...
        return            # ...dann halte sofort!
```

D macht das **Gegenteil** von dem, was P_n auf n tut!

Korollar 6.16: Der Widerspruch

D ist auch ein Programm, also hat D einen Index d in der Aufzählung aller Programme.

Was passiert bei $D(d)$?

- Falls $D(d)$ hält $\Rightarrow H(P_d, d) = 0 \Rightarrow P_d(d)$ loopt $\Rightarrow D(d)$ loopt. **Widerspruch!**
- Falls $D(d)$ loopt $\Rightarrow H(P_d, d) = 1 \Rightarrow P_d(d)$ hält $\Rightarrow D(d)$ hält. **Widerspruch!**

Das ist genau Theorem 6.12! Die "Anti-Diagonale" kann nicht in der Tabelle sein.

Also kann H nicht existieren. $\square$

Zusammenfassung: Ein Theorem, Vier Konsequenzen

Anwendung	Universum U	$P(y, x)$ bedeutet...
Russell's Paradox	Mengen	$y \in x$
Cantor's Diagonal	$\mathbb{N}$	$s_x(y) = 1$
Unberechenbarkeit	$\mathbb{N}$	$P_x(y) = 1$
Halteproblem	$\mathbb{N}$	P_x hält auf y

Das Schöne: Alle diese berühmten Resultate haben dieselbe logische Struktur!

Theorem 6.12 ist die **tiefste Essenz** von Selbstreferenz-Paradoxa.

Übungsblock 3: Diagonalisierung & Selbstreferenz

Aufgabe 1 (HS19-Stil): Zeige mit dem Diagonalargument: Die Menge aller Funktionen $f : \mathbb{N} \to \{0, 1, 2\}$ ist überabzählbar.

Aufgabe 2: Sei $\{P_0, P_1, P_2, ...\}$ eine Aufzählung aller Programme. Definiere $D(n) = 1 - P_n(n)$ (wobei $P_n(n) \in \{0, 1\}$). Zeige: D ist nicht in der Aufzählung, d.h. $D \neq P_k$ für alle k.

Aufgabe 3: Beweise oder widerlege: Die Menge $S = \{A \subseteq \mathbb{N} : A$ oder $\mathbb{N} \setminus A$ ist endlich$\}$ ist abzählbar.

Lösungen Übungsblock 3

Zu Aufgabe 1:

- Angenommen es gäbe Bijektion $f : \mathbb{N} \to (\mathbb{N} \to \{0, 1, 2\})$.
- Schreibe $f_n = f(n)$. Konstruiere $d : \mathbb{N} \to \{0, 1, 2\}$ durch:
- $d(n) = (f_n(n) + 1) \mod 3$.
- Dann: $d(n) \neq f_n(n)$ für alle n, also $d \neq f_n$ für alle n.
- Aber d ist eine Funktion $\mathbb{N} \to \{0, 1, 2\}$. Widerspruch! $\square$

Zu Aufgabe 2:

- Angenommen $D = P_k$ für ein k. Dann:
- $D(k) = 1 - P_k(k) = 1 - D(k)$.
- Also $D(k) = 1 - D(k)$, was unmöglich ist. Widerspruch! $\square$

Lösungen Übungsblock 3 (Fortsetzung)

Zu Aufgabe 3:

- S ist **abzählbar**.
- $S = S_1 \cup S_2$ wobei:
  - $S_1 = \{A \subseteq \mathbb{N} : A \text{ endlich}\}$
  - $S_2 = \{A \subseteq \mathbb{N} : \mathbb{N} \setminus A \text{ endlich}\}$
- S_1 ist abzählbar (endliche Teilmengen von $\mathbb{N}$).
- $S_2 \cong S_1$ via $A \mapsto \mathbb{N} \setminus A$, also auch abzählbar.
- Vereinigung zweier abzählbarer Mengen ist abzählbar. $\square$

Teil 4: Resolution Calculus

Zurück zu den Kalkülen

Letzte Woche: **Syntaktische Ableitung** ($\vdash$) vs. **Semantische Folgerung** ($\models$).

Wir wollen: $M \vdash F$ genau dann, wenn $M \models F$. $\Rightarrow$ Ein **sound** und **complete** Kalkül.

Resolution ist so ein Kalkül für die **Aussagenlogik**.

Motivation: Widerspruchsbeweis

Schlüsselidee: Um $M \models F$ zu zeigen, zeige dass $M \wedge \neg F$ **unerfüllbar** ist.

Warum? Wenn es kein Modell gibt, das M wahr und F falsch macht, dann folgt F aus M.

Resolution: Ein Kalkül, der systematisch Unerfüllbarkeit beweist.

Der Resolution-Kalkül: Setup

Wir arbeiten mit Formeln in **CNF** (Konjunktive Normalform).

Erinnerung (CNF): Eine Formel in CNF ist eine Menge von **Klauseln**. Eine Klausel ist eine Menge von **Literalen** (Atome oder negierte Atome).

Beispiel: $F = (A \vee \neg B) \wedge (\neg A \vee C)$ Als Klauselmenge: $\{\{A, \neg B\}, \{\neg A, C\}\}$

Konvention:

- Klausel = implizites OR zwischen Literalen.
- Formelmenge = implizites AND zwischen Klauseln.
- Leere Klausel $\square$ = Widerspruch (immer falsch).

Definition 6.28: Die Resolutions-Regel

Definition 6.28 (Resolvent): Gegeben zwei Klauseln C_1 und C_2 mit einem Literal $L \in C_1$ und $\neg L \in C_2$:

$$\frac{C_1 \cup \{L\} \quad C_2 \cup \{\neg L\}}{C_1 \cup C_2} \quad \text{(Res)}$$

Die resultierende Klausel heisst **Resolvente** von C_1 und C_2.

Beispiel:

$$\frac{\{A, \neg B\} \quad \{\neg A, C\}}{\{\neg B, C\}}$$

Intuition: Wir "eliminieren" das Literal A, da einer der beiden Fälle (A oder ¬A) wahr sein muss.

Beispiel: Resolution in Aktion

Zeige: $\{A \vee B, \neg A, \neg B\} \models \square$

Klauselmenge: $\{\{A, B\}, \{\neg A\}, \{\neg B\}\}$

Schritt 1: Resolviere $\{A, B\}$ mit $\{\neg A\}$ über A:

$$\frac{\{A, B\} \quad \{\neg A\}}{\{B\}}$$

Schritt 2: Resolviere $\{B\}$ mit $\{\neg B\}$ über B:

$$\frac{\{B\} \quad \{\neg B\}}{\square}$$

Wir haben die **leere Klausel** abgeleitet! $\Longrightarrow$ Widerspruch.

Theorem 6.5: Soundness der Resolution

Theorem (Soundness): Die Resolutions-Regel ist korrekt:

$$\{C_1 \cup \{L\}, C_2 \cup \{\neg L\}\} \models C_1 \cup C_2$$

Beweis: Sei $\mathcal{A}$ ein Modell für beide Klauseln.

- Fall 1: $\mathcal{A}(L) = 1$. Dann muss in $C_2 \cup \{\neg L\}$ ein Literal aus C_2 wahr sein (da $\neg L$ falsch). Also $\mathcal{A}(C_2) = 1$.
- Fall 2: $\mathcal{A}(L) = 0$. Dann muss in $C_1 \cup \{L\}$ ein Literal aus C_1 wahr sein. Also $\mathcal{A}(C_1) = 1$.

In beiden Fällen ist $C_1 \cup C_2$ wahr. $\square$

Theorem 6.6: Completeness der Resolution

Theorem 6.6 (Completeness): Wenn eine Klauselmenge S unerfüllbar ist, dann kann man $\square$ aus S ableiten.

Keine Beweisskizze hier (zu technisch), aber die Intuition ist: Wenn es keinen Widerspruch gibt, kann man ein Modell konstruieren.

Zusammenfassung: Der Resolution-Kalkül ist **sound** und **complete** für die Aussagenlogik.

$$S \text{ unerfüllbar} \iff S \vdash_{\mathsf{Res}} \square$$

Algorithmus: Resolution-Beweis

Input: Klauselmenge S (in CNF). **Output:** "Unerfüllbar" oder "Erfüllbar".

```
1. Wiederhole:
   a. Wähle zwei Klauseln C1, C2 mit L ∈ C1, ¬L ∈ C2.
   b. Berechne Resolvente R = (C1 \ {L}) ∪ (C2 \ {¬L}).
   c. Wenn R = □, gib "Unerfüllbar" aus.
   d. Füge R zu S hinzu, wenn noch nicht vorhanden.
2. Wenn keine neuen Klauseln mehr entstehen,
   gib "Erfüllbar" aus.
```

In der Praxis: SAT-Solver nutzen Heuristiken (DPLL, CDCL).

Beispiel: Tautologie beweisen (Exam-Stil)

Aufgabe: Zeige mit Resolution, dass $(A \to B) \to ((B \to C) \to (A \to C))$ eine Tautologie ist.

Strategie: Eine Formel F ist Tautologie $\iff$ $\neg F$ ist unerfüllbar.

Schritt 1: Negiere die Formel. $\neg F = \neg((A \to B) \to ((B \to C) \to (A \to C)))$
$= (A \to B) \wedge \neg((B \to C) \to (A \to C)) = (A \to B) \wedge (B \to C) \wedge \neg(A \to C)$
$= (A \to B) \wedge (B \to C) \wedge A \wedge \neg C$

Beispiel: Tautologie beweisen (Fortsetzung)

Schritt 2: In CNF umwandeln.

- $A \to B \equiv \neg A \vee B$
- $B \to C \equiv \neg B \vee C$
- A
- $\neg C$

Klauselmenge: $S = \{\{\neg A, B\}, \{\neg B, C\}, \{A\}, \{\neg C\}\}$

Schritt 3: Resolution anwenden.

1. $\{\neg A, B\} + \{A\} \to \{B\}$
2. $\{B\} + \{\neg B, C\} \to \{C\}$
3. $\{C\} + \{\neg C\} \to \square$

Ergebnis: $\neg F$ ist unerfüllbar, also ist F eine Tautologie. ✓

Übungsblock 4: Resolution

Aufgabe 1 (HS19): Sei $F = (A \vee B) \wedge (A \rightarrow \neg C) \wedge \neg(B \wedge C)$ und $G = \neg C$. Zeige mit Resolution: $F \models G$.

Aufgabe 2 (HS18-Stil): Sei $G = (A \vee B) \rightarrow ((B \wedge C) \vee ((A \vee B) \wedge (A \vee \neg C)) \vee (A \wedge B))$. Zeige mit Resolution, dass G eine Tautologie ist.

Aufgabe 3: Finde den Fehler in folgender Resolution: $\{\{A, B\}, \{\neg A, \neg B\}\} \vdash \{A, \neg A\} \vdash \square$

Lösungen Übungsblock 4

Zu Aufgabe 1:

- Zeige: $F \wedge C$ unerfüllbar.
- Klauseln: $\{A, B\}$, $\{\neg A, \neg C\}$, $\{\neg B, \neg C\}$, $\{C\}$
- 1. $\{\neg A, \neg C\} + \{C\} \rightarrow \{\neg A\}$
- 2. $\{\neg B, \neg C\} + \{C\} \rightarrow \{\neg B\}$
- 3. $\{A, B\} + \{\neg A\} \rightarrow \{B\}$
- 4. $\{B\} + \{\neg B\} \rightarrow \square$ ✓

Lösungen Übungsblock 4 (Fortsetzung)

Zu Aufgabe 2: $\neg G$ in CNF: Vereinfache erst $\neg G$, dann CNF. Dies ergibt mehrere Klauseln. Resolution führt zu $\square$. *(Detaillierte Lösung: Siehe Übungsblatt.)*

Zu Aufgabe 3:

- Der Fehler: Man kann nicht über **zwei verschiedene** Literale gleichzeitig resolvieren!
- $\{A, B\}$ und $\{\neg A, \neg B\}$ ergeben entweder:
  - Resolviere über A: $\{B, \neg B\}$ (Tautologie, nutzlos)
  - Resolviere über B: $\{A, \neg A\}$ (Tautologie, nutzlos)
- Tautologische Klauseln können gestrichen werden, führen aber nicht zu $\square$.
- Die ursprüngliche Menge ist erfüllbar ($A = 1, B = 0$)!

Teil 5: Beweispatterns (Cheatsheet)

Die 5 wichtigsten Patterns

Pattern	Ziel	Strategie
1. $F \models G$	Zeige logische Folgerung	$F \wedge \neg G$ unerfüllbar (Resolution)
2. Tautologie	Zeige F immer wahr	$\neg F$ unerfüllbar (Resolution)
3. Semantisch	Zeige $F \models G$ direkt	"Sei $\mathcal{A} \models F$..." Fallunterscheidung
4. Gegenbeispiel	Zeige $F \not\models G$	Finde Modell für $F \wedge \neg G$
5. Äquivalenz	Zeige $F \equiv G$	Beide Richtungen oder Wahrheitstabelle

Pattern-Beispiele (Kurzform)

Pattern 1 (Resolution): $\{A, \neg A \vee B, \neg B \vee C\} \models C \rightarrow$ Füge $\neg C$ hinzu, resolviere zu $\square$.

Pattern 3 (Semantisch): $\forall x(F \vee G) \models F \vee \exists x G \rightarrow$ Fallunterscheidung: $\mathcal{A}(F) = 1$ oder $\mathcal{A}(F) = 0$.

Pattern 4 (Gegenbeispiel): $\exists x P(x) \not\models \forall x P(x) \rightarrow U = \{0, 1\}$, $P = \{0\}$.

Zusammenfassung

Was du mitnehmen solltest

1. **Theorem 6.12:** Die logische Essenz hinter Russell, Cantor, Gödel und Turing. $\neg\exists x \forall y (P(y,x) \iff \neg P(y,y))$ ist eine Tautologie.
2. **Resolution:** Der Kalkül für automatisches Beweisen. Zeige Unerfüllbarkeit durch Ableiten von $\square$.
3. **Prenex Normal Form:** Alle Quantoren nach vorne. Wichtig für Skolemisierung.
4. **Beweispatterns:**
   - $F \models G$: Zeige $F \wedge \neg G$ unerfüllbar.
   - Tautologie: Zeige $\neg F$ unerfüllbar.
   - Semantischer Beweis: Fallunterscheidung über Modelle.

Prüfungstipps

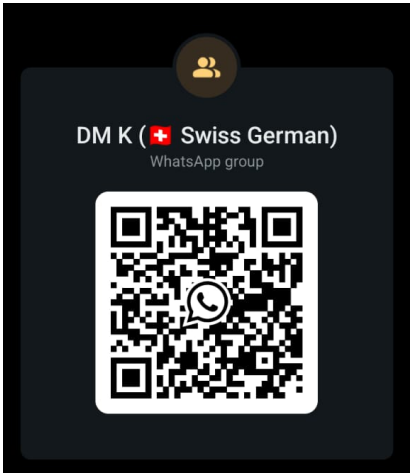
1. **Resolution:** Immer zuerst in CNF bringen, dann systematisch resolvieren.
2. **PNF:** Variablen umbenennen, um Konflikte zu vermeiden!
3. **Semantische Beweise:** Klar angeben, was "Sei $\mathcal{A}$ ein Modell" bedeutet.
4. **Gegenbeispiele:** Konkrete Strukturen angeben (Universum, Interpretation).
5. **Zeit:** Resolution-Aufgaben sind oft mechanisch – nicht zu viel Zeit verlieren.
6. **Theorem 6.12:** Verstehe die Verbindung zu Cantor und Russell!

Die grosse Übersicht: Kapitel 6

Thema	Key Takeaway
Proof Systems	Soundness vs. Completeness
Syntax vs. Semantik	$\vdash$ vs. $\models$
Aussagenlogik	CNF/DNF, Resolution
Prädikatenlogik	Quantoren, Strukturen
Theorem 6.12	Selbstreferenz-Paradoxa
PNF & Skolem	Standardformen
Resolution (PL)	Unifikation

Offene Fragen & Feedback

- Feedback zur heutigen Session?
- E-Mail: dm@shivi.io


DM K (Swiss German)
WhatsApp group