Theoretische Grundlagen der Informatik 3
Logik und Kalküle

Stephan Kreutzer

Wintersemester 2012/2013

1. Einleitung

Einleitung

Inhalt.

Die Vorlesung ist eine Einführung in die Logik im Kontext der Informatik.

Fragen.

1. Was ist überhaupt Logik?
2. Warum wollen Sie das lernen und warum wollen wir, dass Sie das lernen?

Was ist Logik?

Logik. "Logik" kommt vom altgriechischen Wort "logos", die Vernunft.

In der Umgangssprache bezeichnet Logik vernünftiges oder richtiges Schließen oder Denken.

Die Logik hat ihren Ursprung in der (griechischen) Philosophie, in der Frage, was richtiges Argumentieren eigentlich bedeutet, d. h. was man aus einer Aussage folgerichtig ableiten kann.

In dieser Vorlesung beschäftigen wir uns vor allem mit formaler, oder mathematischer Logik.

Logische Sprachen oder Systeme

Im Zentrum der Betrachtungen stehen logische Systeme oder Logiken.

Ein logisches System besteht aus:

Syntax. Eine formal definierte Ansammlung von Formeln. Die Syntax bestimmt, was eine korrekte Formel der Logik sein soll.

Beispiel: $\forall x \exists y \big(x < y \wedge \mathit{prim}(y) \big)$

Semantik. Die Semantik gibt die Bedeutung der einzelnen Formeln an. Zum Beispiel, ob obige Formel in den natürlichen Zahlen $(\mathbb{N}, <, +, *, \mathit{prim})$ gilt.

Vergleiche Programmiersprachen wie C oder Java. Die Syntax gibt an, wann ein Programm korrekt compiliert werden kann. Die Semantik definiert dann, was das Programm eigentlich bedeuten soll.

Anwendungen der Logik in der Informatik

Einige ausgewählte Anwendungen der Logik in der Informatik.

1. Logiken als Abfragesprachen für Datenbanken
2. Logiken als Spezifikationsmechanismen in der automatischen Verifikation von Schaltkreisen, Protokollen und Programmen
3. Logik in der künstlichen Intelligenz
4. Semantik von Programmiersprachen
5. ....

Beispiel: Datenbanken

Relationale Datenbanken

Relationale Datenbanken.

Eines der verbreitetsten Datenmodelle in der Praxis.

Datenbanken bestehen in diesem Modell aus Tabellen bzw. Relationen.

Beispiel. Filmdatenbank wie z.B. IMDB

`Schauspieler`(Vorname, Name, Geburtsjahr)

`Film`(Titel, Regisseur, Entstehungsjahr)

`Cast`(Filmtitel, Schauspieler)

...

Datenbanken

Beispiel.

Schauspieler

Kirstin	Dunst	1982
George	Clooney	1961
...	...	...

Film

Melancholia	Lars von Trier	2011
Good Night, and Good Luck	George Clooney	2007
...	...	...

Cast

Melancholia	Kirsten Dunst	
Good Night, and Good Luck	George Clooney	
...	...	...

Datenbankanfragen. Um Informationen aus Datenbanken zu bekommen stellen wir Datenbankanfragen.

Beispiele. Liste alle Filme von “Lars von Trier” auf.

Gibt es einen Film, in dem George Clooney und Kirstin Dunst mitspielen?

Datenbanken

Datenbankanfragesprachen.

Um solche Anfragen stellen zu können, brauchen wir eine Sprache, in der diese Dinge formuliert werden können.

Wichtige Eigenschaften.

- Anfragen müssen präzise und eindeutig formuliert werden können.

  Natürliche Sprache wegen Doppeldeutigkeiten ungeeignet.

  $\rightsquigarrow$ Logik als Basis von Anfragesprachen.

  Beispiel: SQL = Prädikatenlogik plus Aggregation, Zählen etc.

- In der Logik formulierte Anfragen müssen schnell ausgewertet werden können.

  $\rightsquigarrow$ Auswertungsproblem einer Logik.

Beispiel: Automatische Verifikation

Automatische Verifikation

Schaltkreisverifikation. Eine wichtige industrielle Anwendung der Logik ist die Schaltkreisverifikation.

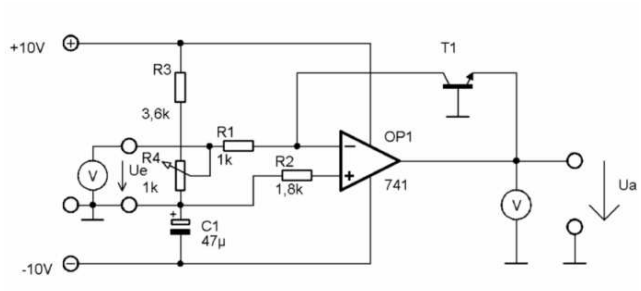
Ziel ist ein Beweis, dass ein gegebener Schaltkreis “korrekt” ist.

Beispiel Flugzeugsicherheitssysteme. “Das Flugzeug darf nie unter 60 Meter Flughöhe sinken, ohne dass das Fahrwerk ausgefahren wird.”

Dazu müssen zunächst die geforderten Eigenschaften des Schaltkreises exakt und formal spezifiziert werden.

Model-Checking

Schaltkreis



Korrektheitseigenschaft

``Das Flugzeug darf nie unter 60 Meter Flughöhe sinken, ohne dass das Fahrwerk ausgefahren wird."

Modell des Schaltkreises *erfüllt?* **Logische Formel**

Wichtige Eigenschaften der verwendeten Logiken.

- Typische Korrektheitseigenschaften müssen formalisierbar sein.
- Formeln müssen effizient auswertbar sein.

Beispiel: Wissensrepräsentation in der künstlichen Intelligenz

Wissensrepräsentation

Ziel. Das Wissen von Experten in bestimmten Bereichen zu speichern und für andere nutzbar und zugänglich zu machen.

Beispiel. Spezialwissen in der Medizin, Biologie, etc.

Wissensrepräsentation. Speicherung von Expertenwissen in Wissensbasen.

Beispiel. Adler $\sqsubseteq$ Vogel $\sqsubseteq$ endothermer Organismus

Dazu werden sogenannte Beschreibungslogiken verwendet.

Wichtige Eigenschaften der verwendeten Logiken.

- Exakte Formalisierung von Wissen möglich.
- Neue Schlüsse aus dem formalisierten Wissen sollen automatisch hergeleitet werden können.

Zusammenfassung

Anwendungen der Logik in der Informatik.

- Logiken werden als Beschreibungs-, Abfrage- oder Spezifikationsformalismen verwendet.
- Wichtige Eigenschaft ist, dass sie exaktes, eindeutiges Formalisieren von Informationen erlauben.
- Verschiedene Anwendungen verlangen oft verschiedenartige Logiken.

  $\rightsquigarrow$ eine Vielzahl untersuchter Logiken in der Informatik.

  Das steht in gewissem Gegensatz zur mathematischen Logik, in der nur wenige Logiken untersucht werden.

Logik in der Informatik

Zentrale Aspekte der Logik in der Informatik.

- Welche Arten von Logiken gibt es?
- Was können wir in einer bestimmten Logik ausdrücken?
- Wie können wir zeigen, dass etwas nicht ausgedrückt werden kann?
- Kann man Aussagen, die in einer bestimmten Logik formalisiert wurden, automatisch beweisen?
- Wie schwer ist es, eine Formel einer bestimmten Logik, in einem Modell auszuwerten?

1.2. Organisatorisches

Organisatorisches

Vorlesung. Donnerstags, 10-12 Uhr MA 001

Materialien.

- Die Vorlesung wird zum Teil auf Folien und zum Teil an der Tafel gehalten.
- Ein vollständiger Foliensatz inklusive des Teils an der Tafel wird online zur Verfügung gestellt.
- Allerdings werden in der Vorlesung auch Beispiele gerechnet und Dinge motiviert, die nicht auf den Folien stehen.
- Prüfungsrelevant ist alles, was in der Vorlesung besprochen wurde (egal ob auf Folie oder nicht).

Literaturhinweise.

- Logik für Informatiker, M. Kreuzer und S. Kühling, Pearson
- Mathematisch-strukturelle Grundlagen der Informatik, H. Ehrig et. al, Springer Verlag
- Einführung in die mathematische Logik, H.-D. Ebbinghaus, J. Flum und W. Thomas, Spektrum

Organisatorisches: Materialien

ISIS-System.

- Vorlesungsmaterialien und Übungsblätter werden über das ISIS-System zur Verfügung gestellt.
- Über dieses System verschicken wir auch Ankündigungen, Sie sollten sich also anmelden.

Foren. Wir haben zwei Nachrichtenforen im ISIS System eingerichtet.

1. Nachrichtenforum für uns, in dem wir Ankündigungen etc. verschicken.
2. Diskussionsforum für Sie, in dem Sie den Stoff der Vorlesung, Organisationsfragen etc. diskutieren können. Wir lesen das unregelmäßig mit,

   d.h. Sie sollten nicht unbedingt darauf vertrauen, dass Sie auf dort gestellte Fragen auch schnell eine Antwort von uns bekommen.

Organisatorisches: Übungen

Übungsblätter.

- Es wird wöchentlich ein Übungsblatt in der Vorlesung ausgeteilt. Beginn: Woche 2.
- Sie haben eine Woche Zeit zur Bearbeitung und geben es dann wieder **nach der Vorlesung** ab.

  Alternativ: bis 10.00 Uhr s.t. im Sekretariat, Raum TEL 711.
- Die Übungen sollen in Gruppen zu maximal 3 Studierenden gelöst werden.
- Es werden 50% der Übungspunkte zur Klausurzulassung benötigt.

Wichtig!!! Auf jedes Übungsblatt müssen Sie Ihren Tutor sowie Tag und Zeit Ihres Tutoriums schreiben!

Wir können Ihre Abgabe sonst nicht zuordnen und nicht korrigieren.

Organisatorisches: Tutorien

Tutorien.

Montag 10-12 Uhr,	EN 187,	Sebastian
Montag 12-14 Uhr,	MA 545,	Friederike
Montag 14-16 Uhr,	H 3008,	Viktor
Montag 16-18 Uhr,	H 3008,	Friederike
Dienstag 10-12 Uhr,	H 3008,	Christoph
Dienstag 14-16 Uhr,	A 060,	Christoph
Dienstag 14-16 Uhr,	MA 144,	Stephan Kreutzer
Dienstag 16-18 Uhr,	EN 185,	Christoph
Mittwoch 10-12 Uhr,	H 3021,	Tobi
Mittwoch 12-14 Uhr,	MA 841,	Tobson
Mittwoch 14-16 Uhr,	H 3008,	Sebastian
Mittwoch 14-16 Uhr,	MA 545,	Viktor

Organisatorisches: Sprechzeiten etc.

Sprechzeiten. Dienstags, 14-15 Uhr, sowie nach Vereinbarung.

Kontakt.

- Raum TEL 712
- EMail: stephan.kreutzer@tu-berlin.de
- Webseite: logic.las.tu-berlin.de

Klausur

Prüfung. Das Modul TheGI 3 wird mit einer Klausur abgeschlossen.

Zur Klausurzulassung müssen Sie 50% der Übungspunkte erreichen.

Ausnahme. Wenn Sie schon bereits eine TheGI 3 Klausur nicht bestanden haben, Sie also eine Widerholungsklausur schreiben, gilt Ihre alte Klausurzulassung weiter.

Klausurtermine.

1. Die Klausur wird am 28.02.2013 stattfinden.
2. Sollten Studierende die Klausur nicht bestehen, werden wir eine Wiederholungsklausur anbieten (oder mündliche Prüfungen).

   Diese würde am 02.04.2013 stattfinden.

2. Aussagenlogik

2.1. Einleitung

Einleitung

Beispiel. Betrachten wir folgende Aussage.

- Wenn der Zug zu spät ist und keine Taxis am Bahnhof stehen, kommt Peter zu spät zu seiner Verabredung.
- Peter kam nicht zu spät zu seiner Verabredung.
- Der Zug hatte Verspätung.

Also standen Taxis am Bahnhof.

Ist das Argument gültig?

Einleitung

Beispiel. Sie kann nicht zu hause sein, da sie entweder an Bord oder zu hause ist und ich gerade gehört habe, dass sie an Bord ist.

Ist das Argument gültig?

Was können wir formal beweisen?

- Klar formulierte Aussagen, die entweder richtig oder falsch sind.

  Die Bedeutung aller verwendeten Ausdrücke muss bekannt sein.

  Ebenso das vorausgesetzte Hintergrundwissen.

- Natürliche Sprache ist dafür nicht gut geeignet.

- Wir werden daher formale Sprachen, oder Logiken, verwenden, in denen alle Ausdrücke formal und vollständig definiert sind.

- Ziel ist es, allgemeine Regeln für korrektes Schließen herleiten zu können, möglichst sogar automatisch.

Beispiel: Syllogismen

Korrektheit folgender Aussagen folgt aus rein formalen Gründen.

Beispiel.

Annahme 1: **Alle** Menschen **sind** sterblich.

Annahme 2: Sokrates **ist ein** Mensch.

Folgerung: **Also ist** Sokrates sterblich.

Diese und verwandte Arten von Schlüssen werden Syllogismen genannt.

Abstrakte Form.

Annahme 1: **Alle** A **sind** B.

Annahme 2: C **ist ein** A.

Folgerung: **Also ist** C B.

Beispiel: Syllogismen

Beispiel.

Annahme 1: **Alle** Ersetzungschiffren **sind** anfällig für brute-force Angriffe.

Annahme 2: Der Caesar Chiffre **ist ein** Ersetzungschiffre.

Folgerung: **Also ist** der Caesar Chiffre anfällig für brute-force Angriffe.

Einleitung

In diesem Teil der Vorlesung werden wir Methoden kennen lernen um

- Aussagen wie auf den vorigen Folien formal auszudrücken
- formalisierte Aussagen zu manipulieren
- logische Behauptungen zu beweisen

2.2. Syntax und Semantik der Aussagenlogik

Propositionen

- Propositionen sind Aussagen die entweder wahr oder falsch sein können

  Beispiel. A : Die Sonne scheint B : Die Vorlesung ist langweilig

- Propositionen können durch Verknüpfungen kombiniert werden, z. B. durch nicht, und, oder or wenn ... dann ....

  Beispiel. A und nicht B

  (Die Sonne scheint und die Vorlesung ist nicht langweilig)

- Die Aussagenlogik studiert grundlegende Prinzipien korrekten Schließens mit Propositionen und deren Kombinationen.

Syntax der Aussagenlogik

Definition 2.1. (Aussagenvariablen)

Eine Aussagenvariable, oder auch einfach Variable, hat die Form V_i für $i \in \mathbb{N}$.

Die Menge aller Variablen bezeichnen wir als AVAR.

Definition 2.2. (Alphabet)

Das Alphabet der Aussagenlogik ist

$$\Sigma_{\mathsf{AL}} := \mathsf{AVAR} \cup \{\top, \bot, \neg, \vee, \wedge, \rightarrow, \leftrightarrow, (,)\}$$

Aussagenlogik

Definition 2.3. (Syntax der Aussagenlogik)

Die Klasse AL der aussagenlogischen Formeln ist induktiv definiert durch

Basis:

- $\top, \bot$ sind aussagenlogische Formeln
- Jede Variable $V_i \in \text{AVAR}$ ist eine aussagenlogische Formel

$\top, \bot$ und die Variablen werden atomare Formeln oder Atome genannt

Induktionsschritt:

- Wenn $\varphi \in \text{AL}$ eine Formel ist, dann auch $\neg\varphi \in \text{AL}$
- Wenn $\varphi, \psi \in \text{AL}$ Formeln sind, dann auch

$$(\varphi \vee \psi), \quad (\varphi \wedge \psi), \quad (\varphi \to \psi), \quad (\varphi \leftrightarrow \psi)$$

$\neg, \vee, \wedge, \to, \leftrightarrow$ werden aussagenlogische Verknüpfungen genannt.

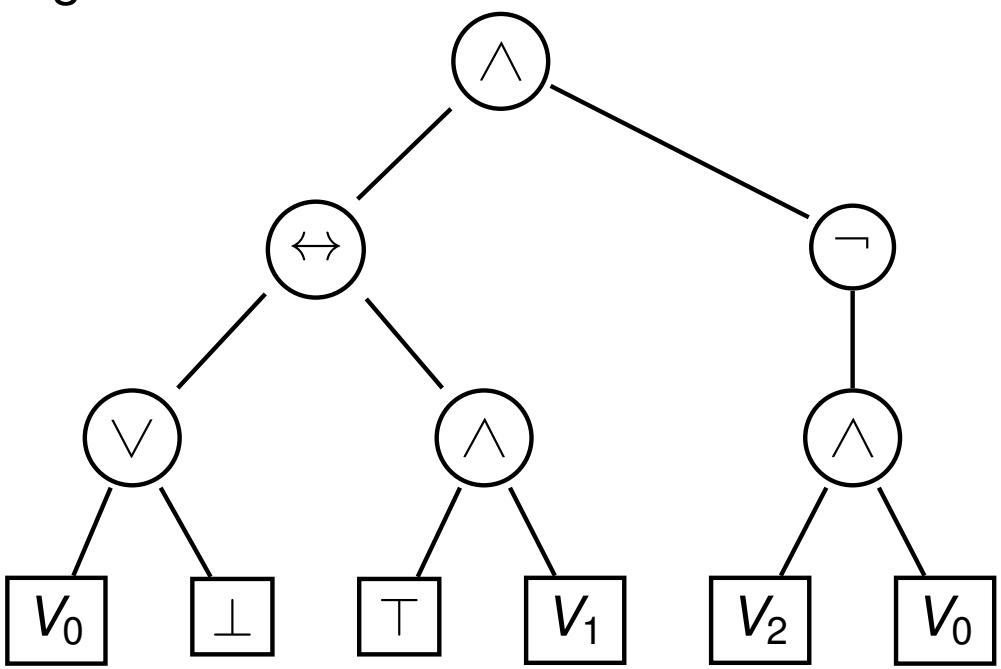
Syntax- oder Ableitungsbäume

Die Struktur einer Formel kann elegant durch ihren Syntax- oder Ableitungsbaum dargestellt werden.

Der Syntaxbaum der Formel

$$\varphi := (((V_0 \vee \bot) \leftrightarrow (\top \wedge V_1)) \wedge \neg(V_2 \wedge V_0))$$

ist definiert wie folgt:



Beispiel

Beispiel 2.4. Erinnern wir uns an das Beispiel vom Anfang der Vorlesung:

- Wenn der Zug zu spät ist und keine Taxis am Bahnhof stehen, kommt Peter zu spät zu seiner Verabredung.
- Peter kam nicht zu spät zu seiner Verabredung.
- Der Zug hatte Verspätung.

Also standen Taxis am Bahnhof.

Aussagenvariablen:

T: der Zug war zu spät
C: es standen Taxis am Bahnhof
L: Peter kam zu spät zu seiner Verabredung

Formel: $((((((T \wedge \neg C) \rightarrow L) \wedge \neg L) \wedge T) \rightarrow C)$

Semantik der Aussagenlogik

Aussagenvariablen:

T: der Zug war zu spät
C: es standen Taxis am Bahnhof
L: Peter kam zu spät zu seiner Verabredung

Formel: $((((((T \wedge \neg C) \rightarrow L) \wedge \neg L) \wedge T) \rightarrow C)$

- Ist die Aussage und somit die Formel wahr?
- Was bedeutet es, wenn eine Formel wahr oder falsch ist?

Um diese Fragen formal zu beantworten, müssen wir für aussagenlogische Formeln eine formale Semantik definieren.

Einschub: Unterformeln und induktive Definitionen

Unterformeln

Definition 2.5.

Die Menge $\mathrm{sub}(\varphi)$ der Unterformeln einer Formel φ ist induktiv wie folgt definiert:

- Ist φ atomar, dann ist $\mathrm{sub}(\varphi) := \{\varphi\}$
- Ist $\varphi := \neg\psi$, dann ist $\mathrm{sub}(\varphi) := \{\varphi\} \cup \mathrm{sub}(\psi)$
- Für alle $* \in \{\wedge, \vee, \rightarrow, \leftrightarrow\}$: Ist $\varphi := (\varphi_1 * \varphi_2)$, dann ist

$$\mathrm{sub}(\varphi) := \{\varphi\} \cup \mathrm{sub}(\varphi_1) \cup \mathrm{sub}(\varphi_2)$$

Bemerkung 2.6.

Formal ist sub eine Funktion

$$\mathrm{sub} : \mathrm{AL} \rightarrow \mathcal{P}(\mathrm{AL})$$

Beispiel

Formel: $\varphi := ((((((T \wedge \neg C) \rightarrow L) \wedge \neg L) \wedge T) \rightarrow C)$

Dann ist $\text{sub}(\varphi) := \{$ $((((((T \wedge \neg C) \rightarrow L) \wedge \neg L) \wedge T) \rightarrow C)$,
$(((T \wedge \neg C) \rightarrow L) \wedge \neg L) \wedge T)$,
C,
$((T \wedge \neg C) \rightarrow L) \wedge \neg L)$,
$((T \wedge \neg C) \rightarrow L)$,
$\neg L$,
L,
$(T \wedge \neg C)$,
T,
$\neg C\}$

Induktive Definition über Formeln

Die Definition der Menge der Unterformeln ist ein Beispiel für eine sehr elegante Methode, Funktionen über Formeln zu definieren oder Aussagen über Formeln zu beweisen.

Induktive Definitionen.

Eine Funktion $f : \mathsf{AL} \to M$, für eine beliebige Menge M, kann induktiv wie folgt definiert werden:

Basisfälle:

- Definiere $f(\top)$ und $f(\bot)$
- Definiere $f(X)$ für alle $X \in \mathsf{AVAR}$

Induktionsschritt:

- Definiere $f(\neg\varphi)$ aus $f(\varphi)$
- Für $* \in \{\wedge, \vee, \to, \leftrightarrow\}$ definiere $f((\varphi * \psi))$ aus $f(\varphi)$ und $f(\psi)$

Beispiel

Wir wollen eine Funktion $f : \mathrm{AL} \to \mathbb{N}$ definieren, so dass $f(\varphi)$ die "Größe" der Formel φ angibt.

Basisfälle:

- *Definiere* $f(\top)$ *und* $f(\bot)$
  $f(\top) = f(\bot) := 1$
- *Definiere* $f(X)$ *für alle* $X \in \mathrm{AVAR}$
  $f(X) := 1$ für alle $X \in \mathrm{AVAR}$

Induktionsschritt:

- *Definiere* $f(\neg\varphi)$ *aus* $f(\varphi)$
  $f(\neg\varphi) := 1 + f(\varphi)$
- *Für* $* \in \{\wedge, \vee, \to, \leftrightarrow\}$ *definiere* $f((\varphi * \psi))$ *aus* $f(\varphi)$ *und* $f(\psi)$
  $f\big((\varphi * \psi)\big) := 1 + f(\varphi) + f(\psi)$

Semantik der Aussagenlogik

Wahrheitsbelegungen

Definition 2.7. Die Menge $\mathrm{var}(\varphi)$ der Variablen einer Formel φ ist die Menge

$$\mathrm{var}(\varphi) := \mathrm{AVAR} \cap \mathrm{sub}(\varphi)$$

Definition 2.8.

1. Eine Wahrheitsbelegung, oder kurz Belegung, ist eine partielle Funktion
$$\beta : \mathrm{AVAR} \to \{0, 1\}.$$
2. Eine Belegung β ist eine Belegung für eine Formel φ, oder ist passend für φ, wenn $\mathrm{var}(\varphi) \subseteq \mathrm{Dom}(\beta)$.

Intuitiv: 1 steht für *wahr* und 0 für *falsch*.

Semantik der Aussagenlogik: Intuitive Bedeutung

Atome:

$\top, \bot$ stehen für *wahr* und *falsch*

Variablen V_i stehen für Aussagen. Wir interessieren uns allerdings nur dafür, ob sie wahr oder falsch sind.

Der Wahrheitswert wird durch die Belegung angegeben.

Negation: Ist $\neg\varphi$ wahr, so ist φ falsch.

Also steht $\neg$ für "nicht"

Konjunktion: Ist $(\varphi \wedge \psi)$ wahr so ist φ wahr und ψ wahr

$\wedge$ bedeutet "und"

Disjunktion: $\vee$ bedeutet "oder"

Implikation: $(\varphi \rightarrow \psi)$ bedeutet "φ impliziert ψ" oder wenn φ dann ψ

Biimplikation: $(\varphi \leftrightarrow \psi)$ bedeutet "φ dann, und nur dann, wenn ψ"

Semantik der Aussagenlogik

Definition 2.9. Per Induktion über die Struktur der Formeln in AL definieren wir eine Funktion $[\![\cdot]\!]^{\cdot}$, die jeder Formel $\varphi \in \mathrm{AL}$ und jeder zu φ passenden Belegung β einen Wahrheitswert $[\![\varphi]\!]^{\beta} \in \{0, 1\}$ zuordnet.

Basisfall.

- $[\![\bot]\!]^{\beta} := 0 \qquad [\![\top]\!]^{\beta} := 1$
- Für alle $X \in \mathrm{AVAR}$ gilt $[\![X]\!]^{\beta} := \beta(X)$

Induktionsschritt. Für zusammen gesetzte Formeln φ definieren wir $[\![\varphi]\!]^{\beta}$ durch die folgenden Wahrheitstafeln:

$[\![\varphi]\!]^{\beta}$	$[\![\neg\varphi]\!]^{\beta}$
0	1
1	0

Semantik der Aussagenlogik

Konjunktion

$[\![\varphi]\!]^\beta$	$[\![\psi]\!]^\beta$	$[\![(\varphi \wedge \psi)]\!]^\beta$
0	0	0
0	1	0
1	0	0
1	1	1

Disjunktion

$[\![\varphi]\!]^\beta$	$[\![\psi]\!]^\beta$	$[\![(\varphi \vee \psi)]\!]^\beta$
0	0	0
0	1	1
1	0	1
1	1	1

Implikation

$[\![\varphi]\!]^\beta$	$[\![\psi]\!]^\beta$	$[\![(\varphi \rightarrow \psi)]\!]^\beta$
0	0	1
0	1	1
1	0	0
1	1	1

Biimplikation

$[\![\varphi]\!]^\beta$	$[\![\psi]\!]^\beta$	$[\![(\varphi \leftrightarrow \psi)]\!]^\beta$
0	0	1
0	1	0
1	0	0
1	1	1

Eine Bemerkung zur Implikation

Bemerkung. Logische Implikation $\rightarrow$ stimmt nicht immer mit der umgangssprachlichen Verwendung der Implikation überein.

Zum Beispiel wird keine Kausalität impliziert.

$\varphi \rightarrow \psi$ heißt einfach, dass wann immer φ wahr ist, so muss auch ψ wahr sein.

Insbesondere, wenn φ falsch ist, dann ist $\varphi \rightarrow \psi$ als Aussage wahr.

Über die Wahl der Verknüpfungen

Unsere Wahl der Verknüpfungen $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$ für die Aussagenlogik spiegelt unsere Verwendung in natürlicher Sprache wieder, ist aber zu bestimmtem Grad willkürlich.

Durch die Definition einer Wahrheitstafel können wir auch andere Verknüpfungen und somit auch andere "Aussagenlogiken" definieren.

Beispiel. Exklusives Oder $\varphi \oplus \psi$

Exklusives Oder

$[\![\varphi]\!]^\beta$	$[\![\psi]\!]^\beta$	$[\![\varphi \oplus \psi]\!]^\beta$
0	0	0
0	1	1
1	0	1
1	1	0

Intuitiv: $\varphi \oplus \psi$ bedeutet "entweder φ oder ψ"

Notation

Notation. Wir vereinbaren folgende Notation.

- Belegungen: $\beta, \gamma, ...$
- Formeln: $\varphi, \psi, \varphi'...$
- Mengen von Formeln: $\Phi, \Psi, ...$
- Wir werden auch $X, Y, ...$ für Variablen verwenden

Präferenzregeln. Um unnötige Klammern zu vermeiden,

- lassen wir die äußersten Klammern weg
- vereinbaren, dass $\neg$ stärker bindet als die anderen Verknüpfungen
- $\wedge, \vee$ bindet stärker als $\rightarrow, \leftrightarrow$

Wir schreiben also $\neg X \wedge Y \rightarrow T$ für $((\neg X \wedge Y) \rightarrow T)$

Aber wir können nicht $X \wedge Y \vee Z$ schreiben.

Formalisierung natürlichsprachlicher Aussagen

Beispiel 2.10.

"Wenn es kalt ist und regnet, gehe ich nicht spazieren"

Atomare Aussagen.

- X_k: es ist kalt
- X_r: es regnet
- X_s: ich gehe spazieren

Obige Aussage kann also wie folgt formalisiert werden

$$X_k \wedge X_r \rightarrow \neg X_s$$

Formalisierung natürlichsprachlicher Aussagen

Beispiel 2.11.

"Das Fluchtauto war rot oder grün und hatte vorne und hinten keine Nummernschilder"

Atomare Aussagen.

- X_R: das Auto war rot
- X_G: das Auto war grün
- X_V: das Auto hatte Nummernschilder vorne
- X_H: das Auto hatte Nummernschilder hinten

Zusammengesetze Aussage.

$$(X_R \vee X_G) \wedge \neg X_V \wedge \neg X_H$$

Formalisierung natürlichsprachlicher Aussagen

Das folgende Beispiel ist nicht rein "aussagenlogisch". Eine Formalisierung in der Aussagenlogik kann dennoch nützlich sein.

Beispiel 2.12.

Annahme 1: **Alle** Menschen **sind** sterblich.

Annahme 2: Sokrates **ist ein** Mensch.

Folgerung: **Also ist** Sokrates sterblich.

Atomare Aussagen.

- X_M: x ist ein Mensch
- X_S: x ist sterblich
- X_K: x ist Sokrates

Zusammengesetzte Aussage. $\big((X_M \to X_S) \wedge (X_K \to X_M)\big) \to (X_K \to X_S)$

Dies vernachlässigt aber, dass die Aussage für alle Menschen gelten soll. $\rightsquigarrow$ Prädikatenlogik

Formalisierung natürlichsprachlicher Aussagen

Beispiel

Ein Anhänger einer Sportmannschaft B stellt in einem Turnier folgende Überlegungen an.

- Wenn B gegen M gewinnt, dann gewinnt B das Turnier.
- Wenn B gegen A gewinnt und A gegen C verliert, dann gewinnt B das Turnier.
- Gewinnt M gegen A und C und veliert B nicht gegen A, dann gewinnt B das Turnier.

Atomare Aussagen.

$X_{B,M}$: B gewinnt gegen M $X_{B,A}$: B gewinnt gegen A
$X_{A,B}$: A gewinnt gegen B $X_{B,C}$: B gewinnt gegen C
$X_{A,C}$: A gewinnt gegen C $X_{M,A}$: M gewinnt gegen A
$X_{M,C}$: M gewinnt gegen C X_B: B gewinnt das Turnier

Zusammengesetzte Aussage.

$$(X_{B,M} \to X_B) \wedge (X_{B,A} \wedge \neg X_{A,C} \to X_B) \wedge (X_{M,A} \wedge X_{M,C} \wedge \neg X_{A,B} \to X_B)$$

Formalisierung natürlichsprachlicher Aussagen

Beispiel 2.13.

Annahme 1: **Alle** Menschen **sind** sterblich.

Annahme 2: Sokrates **ist ein** Mensch.

Folgerung: **Also ist** Sokrates sterblich.

Atomare Aussagen.

- X_M: x ist ein Mensch
- X_S: x ist sterblich
- X_K: x ist Sokrates

Zusammengesetzte Aussage. $\big((X_M \to X_S) \wedge (X_K \to X_M)\big) \to (X_K \to X_S)$

Wie hilft uns das zu entscheiden, ob unsere Folgerung korrekt war?

2.3. Erfüllbarkeit und Allgemeingültigkeit

Erfüllbarkeit und Gültigkeit

Definition 2.14. Sei $\varphi \in \mathrm{AL}$ eine Formel.

1. Eine zu φ passende Belegung β erfüllt φ, oder ist ein Modell von φ, wenn $[\![\varphi]\!]^\beta = 1$.

   Wir schreiben $\beta \models \varphi$.

2. φ ist erfüllbar, wenn es eine Belegung β gibt, die φ erfüllt. Anderenfalls ist φ unerfüllbar.

3. φ ist allgemeingültig, oder eine Tautologie, wenn jede zu φ passende Belegung φ erfüllt.

Beobachtung 2.15. Für alle Formeln $\varphi \in \mathrm{AL}$:

φ ist allgemeingültig genau dann, wenn $\neg\varphi$ unerfüllbar ist.

Erfüllbarkeit und Gültigkeit

Atomare Aussagen.

- X_M: x ist ein Mensch
- X_S: x ist sterblich
- X_K: x ist Sokrates

Zusammengesetzte Aussage. $\varphi := \big((X_M \to X_S) \wedge (X_K \to X_M)\big) \to (X_K \to X_S)$

Korrektheit von Aussagen.

Die Folgerung ist also genau dann korrekt, wenn φ allgemeingültig ist.

Wir brauchen also Methoden, um Allgemeingültigkeit und Erfüllbarkeit von Formeln zu entscheiden.

In dieser Vorlesung werden wir drei Methoden kennen lernen

- Wahrheitstafeln
- Resolution
- Sequenzenkalkül

Wahrheitstafeln

Wir können die Wahrheitswerte aussagenlogischer Formeln $\varphi \in AL$ unter allen möglichen Belegungen in einer Wahrheitstafel darstellen.

Für jede mögliche Belegung $\beta : var(\varphi) \to \{0, 1\}$ hat die Tafel eine Zeile die den Wahrheitswert $\beta(X)$ für alle $X \in var(\varphi)$ sowie $[\![\varphi]\!]^\beta$ enthält.

Beispiel 2.16. $\big((X_H \to X_M) \wedge (X_S \to X_H)\big) \to (X_S \to X_M)$

X_H	X_M	X_S	$\big((X_H \to X_M) \wedge (X_S \to X_H)\big) \to (X_S \to X_M)$
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

Erweiterte Wahrheitstafeln

Es ist oft nützlich, die Wahrheitswerte der Unterformeln ebenfalls in der Wahrheitstafel aufzuführen.

Beispiel (forts.) $\varphi := \big((X_H \to X_M) \wedge (X_S \to X_H)\big) \to (X_S \to X_M)$

X_H	X_M	X_S	$(X_H \to X_M)$	$(X_S \to X_H)$	$(X_H \to X_M) \wedge (X_S \to X_H)$	$(X_S \to X_M)$	φ
0	0	0	1	1	1	1	1
0	0	1	1	0	0	0	1
0	1	0	1	1	1	1	1
0	1	1	1	0	0	1	1
1	0	0	0	1	0	1	1
1	0	1	0	1	0	0	1
1	1	0	1	1	1	1	1
1	1	1	1	1	1	1	1

Wir nennen das die erweiterte Wahrheitstafel.

Wahrheitstafel

Bemerkung 2.17. Wenn die Formel φ keine Variable enthält, so besteht die Wahrheitstafel aus einer einzigen Zeile.

Sei $\varphi := (\top \to \bot) \to \bot$

$$\frac{(\top \to \bot) \to \bot}{\top}$$

Die erweiterte Wahrheitstafel ist

$\bot$	$\top$	$\top \to \bot$	$(\top \to \bot) \to \bot$
$\bot$	$\top$	$\bot$	$\top$

Das Koinzidenz Lemma

Wenn wir mit Wahrheitstafeln arbeiten, nehmen wir implizit an, dass $[\![\varphi]\!]^\beta$ nur von den Belegungen der Variablen in $\mathrm{var}(\varphi)$ abhängt, aber nicht von der Belegung anderer Variablen.

Eine Rechtfertigung dieser Annahme liefert das folgende Lemma.

Lemma 2.18. (Koinzidenzlemma)

Sei $\varphi \in \mathrm{AL}$ eine Formel und seien β, β' Belegungen so dass

$$\beta(X) = \beta'(X) \qquad \text{für alle } X \in \mathrm{var}(\varphi).$$

Dann gilt $[\![\varphi]\!]^\beta = [\![\varphi]\!]^{\beta'}$.

Intuitiv ist das Lemma klar, da die anderen Variablen $Y \notin \mathrm{var}(\varphi)$ nicht in der Definition von $[\![\varphi]\!]^\beta$ auftauchen.

Formal kann das Lemma durch strukturelle Induktion bewiesen werden (Übung).

Syllogismen

Syllogismenbeispiel.

- X_M: x ist ein Mensch
- X_S: x ist sterblich
- X_K: x ist Sokrates

Zusammengesetzte Aussage. $\varphi := \big((X_M \to X_S) \wedge (X_K \to X_M)\big) \to (X_K \to X_S)$

Wahrheitstafel.

X_H	X_M	X_S	$\big((X_H \to X_M) \wedge (X_S \to X_H)\big) \to (X_S \to X_M)$
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

Folgerung. Diese Form der Syllogismen ist also korrekt.

Das Wahrheitstafelverfahren

Beobachtung 2.19. Sei $\varphi \in \mathrm{AL}$ eine Formel.

1. φ ist genau dann erfüllbar, wenn die letzte Spalte der Wahrheitstafel mindestens eine 1 enthält.
2. φ ist genau dann unerfüllbar, wenn alle Einträge der letzten Spalte 0 sind.
3. φ ist genau dann allgemeingültig, wenn alle Einträge der letzten Spalte 1 sind.

Das Wahrheitstafelverfahren.

Eingabe: Eine Formel $\varphi \in \mathrm{AL}$.

Ziel: entscheide, ob φ erfüllbar ist.

Methode:
1. Berechne die Wahrheitstafel für φ.
2. Überprüfe, ob die letzte Spalte eine 1 enthält.

Bemerkung. Für Allgemeingültigkeit entscheide, ob die letzte Spalte nur 1 enthält.

Effizienz des Wahrheitstafelverfahrens

Die Wahrheitstafel einer Formel mit n Variablen hat 2^n Zeilen.

Das macht das Wahrheitstafelverfahren extrem ineffizient außer für sehr kleine Formeln.

Variablen	Zeilen
10	$1{,}024 \approx 10^3$
20	$1{,}048{,}576 \approx 10^6$
30	$1{,}073{,}741{,}824 \approx 10^9$
40	$1{,}099{,}511{,}627{,}776 \approx 10^{12}$
50	$1{,}125{,}899{,}906{,}842{,}624 \approx 10^{15}$
60	$1{,}152{,}921{,}504{,}606{,}846{,}976 \approx 10^{18}$

Das Aussagenlogische Erfüllbarkeitsproblem

Bemerkung.

- Das Erfüllbarkeitsproblem der Aussagenlogik ist eines der am besten studierten Probleme der Informatik.
- Es ist “schwer” zu lösen (NP-vollständig, werden wir später beweisen)
- Allerdings existieren Verfahren, die das Problem für viele in der Praxis vorkommende Formeln sehr effizient lösen können. (Das Wahrheitstafelverfahren gehört nicht dazu)
- Diese haben wichtige Anwendungen in der Informatik, z.B. in der Verifikation.

2.4. Logische Äquivalenz und Normalformen

Normalformen

In diesem Abschnitt werden wir Methoden behandeln, Formeln umzuformen, ohne den Wahrheitswert der Formeln zu ändern.

Dies werden wir insbesondere für sogenannte Normalformen benutzen.

Normalformen. Allgemein ist eine Normalform eine Klasse aussagenlogischer Formeln, so dass jede Formel in AL zu einer Formel in dieser Normalform äquivalent ist.

Dazu werden wir zunächst den Äquivalenzbegriff von Formeln formal definieren und einige nützliche Äquivalenzen zwischen Formeln einführen.

Äquivalenz von Formeln

Definition 2.20. Zwei Formeln $\varphi, \psi \in \mathrm{AL}$ sind äquivalent, geschrieben $\varphi \equiv \psi$, wenn für alle Belegungen β gilt:

$$\beta \models \varphi \iff \beta \models \psi$$

Proposition 2.21.

1. Für alle Formeln φ, ψ

$$\varphi \equiv \psi \qquad \Longleftrightarrow \qquad \varphi \leftrightarrow \psi \text{ ist allgemeingültig.}$$

2. Für alle Formeln φ,

$$\varphi \text{ ist allgemeingültig} \iff \varphi \equiv \top.$$

Beispiel

Beispiel 2.22. Für alle $X, Y \in \mathrm{AVAR}$:

$$(X \to Y) \equiv \neg X \vee Y$$
$$(X \leftrightarrow Y) \equiv (X \to Y) \wedge (Y \to X)$$

Beweis mittels Wahrheitstafeln.

$[\![X]\!]^\beta$	$[\![Y]\!]^\beta$	$[\![X \to Y]\!]^\beta$	$[\![\neg X \vee Y]\!]^\beta$	$[\![(X \leftrightarrow Y)]\!]^\beta$	$[\![(X \to Y) \wedge (Y \to X)]\!]^\beta$
0	0	1	1	1	1
0	1	1	1	0	0
1	0	0	0	0	0
1	1	1	1	1	1

Frage. Können wir aus der Äquivalenz $(X \to Y) \equiv \neg X \vee Y$, für alle $X, Y \in \mathrm{AVAR}$, schließen, dass für alle $\varphi, \psi \in \mathrm{AL}$

$$(\varphi \to \psi) \equiv \neg\varphi \vee \psi?$$

Das Substitutionslemma (intuitiv)

Lemma 2.23. Wenn wir in einer Äquivalenz alle Vorkommen einer von Variablen $X_1, \ldots, X_n$ durch Formeln $\varphi_1, \ldots, \varphi_n$ ersetzen, dann bleibt die Äquivalenz erhalten.

Korollar 2.24. Für alle Formeln $\varphi, \psi \in \mathsf{AL}$ gilt:

$$\begin{aligned} (\varphi \to \psi) &\equiv \neg\varphi \vee \psi \\ (\varphi \leftrightarrow \psi) &\equiv (\varphi \to \psi) \wedge (\psi \to \varphi) \end{aligned}$$

Definition 2.25. Eine Substitution ist eine partielle Abbildung

$$\mathcal{S} : \mathsf{AVAR} \to \mathsf{AL}$$

von Aussagenvariablen auf aussagenlogische Formeln mit endlichem Definitionsbereich, d.h. $\mathcal{S}$ ist nur für endlich viele Variablen definiert.

Substitution

Für eine Formel $\varphi \in \mathrm{AL}$ und eine Substitution $\mathcal{S}$ schreiben wir $\varphi\mathcal{S}$ für die Formel, die wir aus φ erhalten, wenn alle Vorkommen einer Variable $X \in \mathrm{Dom}(\mathcal{S})$ in φ durch die Formeln $\mathcal{S}(X)$ ersetzt werden.

Definition 2.26. Für jede Formel $\varphi \in \mathrm{AL}$ und Substitution $\mathcal{S}$ definieren wir die Formel $\varphi\mathcal{S} \in \mathrm{AL}$ induktiv wie folgt:

Induktionsbasis.

- $\bot\mathcal{S} := \bot \qquad \top\mathcal{S} := \top$
- Für $X \in \mathrm{AVAR}$ definieren wir $X\mathcal{S} := \begin{cases} \mathcal{S}(X) & \text{wenn } X \in \mathrm{Dom}(\mathcal{S}) \\ X & \text{sonst} \end{cases}$

Induktionsschritt.

- $(\neg\varphi)\mathcal{S} := \neg(\varphi\mathcal{S})$
- Für $* \in \{\wedge, \vee, \rightarrow, \leftrightarrow\}$ definieren wir

$$(\varphi * \psi)\mathcal{S} := \varphi\mathcal{S} \ * \ \psi\mathcal{S}.$$

Beispiel

Sei $\varphi := V_1 \wedge V_2 \rightarrow V_1 \vee V_3$ und sei

$$\mathcal{S} : \{V_1, V_3\} \rightarrow \mathsf{AL}$$

definiert als $\mathcal{S}(V_1) := V_2$ und $\mathcal{S}(V_2) := (V_0 \vee V_1)$.

Dann gilt

$$\begin{aligned} \varphi\mathcal{S} &= (V_1 \wedge V_2)\mathcal{S} \rightarrow (V_1 \vee V_3)\mathcal{S} \\ &= V_1\mathcal{S} \wedge V_2\mathcal{S} \rightarrow V_1\mathcal{S} \vee V_3\mathcal{S} \\ &= V_2 \wedge (V_0 \vee V_1) \rightarrow V_2 \vee V_3 \end{aligned}$$

Beispiel

Sei $\varphi := ((V_1 \vee V_2) \wedge \neg(V_3 \wedge V_1))$ und sei $\mathcal{S} : \{V_1, V_2, V_3\} \to \mathsf{AL}$

definiert als $\mathcal{S} : \begin{cases} V_1 & \mapsto \neg(V_2 \wedge V_3) \\ V_2 & \mapsto (V_4 \vee V_5) \\ V_3 & \mapsto (V_2 \wedge V_3) \end{cases}$

Dann gilt

$$\begin{aligned} \varphi\mathcal{S} &= ((V_1 \vee V_2) \wedge \neg(V_3 \wedge V_1))\mathcal{S} \\ &\quad ((V_1 \vee V_2)\mathcal{S} \wedge \neg(V_3 \wedge V_1)\mathcal{S}) \\ &\quad ((V_1 \vee V_2)\mathcal{S} \wedge (\neg(V_3 \wedge V_1))\mathcal{S}) \\ &\quad ((V_1\mathcal{S} \vee V_2\mathcal{S}) \wedge \neg((V_3 \wedge V_1)\mathcal{S})) \\ &\quad ((V_1\mathcal{S} \vee V_2\mathcal{S}) \wedge \neg(V_3\mathcal{S} \wedge V_1\mathcal{S})) \\ &\quad ((\neg(V_2 \wedge V_3) \vee (V_4 \vee V_5)) \wedge \neg((V_2 \wedge V_3) \wedge \neg(V_2 \wedge V_3))) \end{aligned}$$

Das Substitutionslemma (formal)

Lemma 2.27. Sei $\mathcal{S}$ eine Substitution und seien $\varphi, \varphi' \in \mathrm{AL}$ Formeln. Dann gilt

$$\varphi \equiv \varphi' \quad \Rightarrow \quad \varphi\mathcal{S} \equiv \varphi'\mathcal{S}.$$

Beweis. Der Beweis wird per Induktion über den Formelaufbau geführt.

Wir führen strukturelle Induktion zunächst allgemein ein und beweisen dann das Lemma.

Strukturelle Induktion

- Beweise über strukturelle Induktion basieren auf dem induktiven Aufbau aussagenlogischer Formeln.
- Das Prinzip ist eine elegante und nützliche Methode, Eigenschaften aussagenlogischer Formeln zu beweisen.

Um zu beweisen, dass eine Eigenschaft A für alle Formeln der Aussagenlogik gilt, müssen wir folgende Aussagen zeigen:

Induktionsbasis. Alle atomare Formeln φ haben die Eigenschaft A.

Induktionsschritt.

- Wenn φ die Eigenschaft A hat, so auch $\neg\varphi$.
- Wenn ψ und φ die Eigenschaft A haben, dann gilt A auch für $(\varphi * \psi)$, mit $* \in \{\vee, \wedge, \rightarrow, \leftrightarrow\}$.

Dann gilt A für alle Formeln $\varphi \in \mathrm{AL}$.

Beweis des Substitutionslemmas

Um das Substitutionslemma zu beweisen, brauchen wir zunächst noch etwas Notation.

Sei $\mathcal{S}$ eine Substitution.

- Eine Belegung β ist passend für $\mathcal{S}$, wenn sie passend für alle Formeln $\mathcal{S}(X)$ mit $X \in \mathrm{Dom}(\mathcal{S})$ ist.
- Ist β eine zu $\mathcal{S}$ passende Belegung, so definieren wir $\beta\mathcal{S}$ wie folgt

$$\beta\mathcal{S}(X) := \begin{cases} [\![\mathcal{S}(X)]\!]^{\beta} & \text{wenn } X \in \mathrm{Dom}(\mathcal{S}) \\ \beta(X) & \text{wenn } X \in \mathrm{Dom}(\beta) \setminus \mathrm{Dom}(\mathcal{S}) \end{cases}$$

Lemma 2.27-A. Für alle Formeln φ und alle zu φ und $\mathcal{S}$ passenden Belegungen β gilt:

$$\beta \models \varphi\mathcal{S} \quad \Longleftrightarrow \quad \beta\mathcal{S} \models \varphi$$

Beweis des Substitutionslemmas

Lemma 2.27-A. Für alle Formeln φ und alle zu φ und $\mathcal{S}$ passenden Belegungen β gilt:

$$\beta \models \varphi\mathcal{S} \quad \Longleftrightarrow \quad \beta\mathcal{S} \models \varphi$$

Wir beweisen das Lemma per Induktion über den Formelaufbau.

Induktionsbasis.

- Für $\varphi \in \{\top, \bot\}$ gilt $\varphi\mathcal{S} = \varphi$
- Für $\varphi := X$, wobei $X \in \text{AVAR} \setminus \text{Dom}(\mathcal{S})$, gilt:
  $\beta\mathcal{S}(X) = \beta(X)$ und $\varphi = \varphi\mathcal{S}$ und daher $[\![\varphi\mathcal{S}]\!]^{\beta} = [\![\varphi]\!]^{\beta\mathcal{S}}$.
- Für $\varphi := X \in \text{Dom}(\mathcal{S})$ gilt: $\varphi\mathcal{S} = \mathcal{S}(X)$ und $\beta\mathcal{S}(X) = [\![\mathcal{S}(X)]\!]^{\beta}$.
  Somit, $\beta \models \varphi\mathcal{S} \quad \Longleftrightarrow \quad \beta\mathcal{S} \models \varphi$.

Beweis des Substitutionslemmas

Inductionsschritt.

- Negation.

$$\begin{aligned}
\beta \models (\neg\varphi)\mathcal{S} &\iff \beta \models \neg(\varphi\mathcal{S}) && \text{Def. der Substitution}\\
&\iff \beta \not\models \varphi\mathcal{S}\\
&\iff \beta\mathcal{S} \not\models \varphi && \text{Induktionsvoraussetzung}\\
&\iff \beta\mathcal{S} \models \neg\varphi.
\end{aligned}$$

- Konjunktion.

$$\begin{aligned}
\beta \models (\varphi \wedge \psi)\mathcal{S} &\iff \beta \models (\varphi\mathcal{S} \wedge \psi\mathcal{S}) && \text{Def. der Subst.}\\
&\iff \beta \models \varphi\mathcal{S} \text{ und } \beta \models \psi\mathcal{S}\\
&\iff \beta\mathcal{S} \models \varphi \text{ und } \beta\mathcal{S} \models \psi && \text{Ind. Vor.}\\
&\iff \beta\mathcal{S} \models (\varphi \wedge \psi).
\end{aligned}$$

- Das Argument für $* \in \{\vee, \rightarrow, \leftrightarrow\}$ ist analog.

Beweis des Substitutionslemmas

Mit Lemma 2.27-A können wir nun das Substitutionslemma beweisen.

Seien φ, φ' äquivalente Formeln.

Wir zeigen, dass $\varphi\mathcal{S} \equiv \varphi'\mathcal{S}$, d.h. das für alle passenden Belegungen β:

$$\beta \models \varphi\mathcal{S} \iff \beta \models \varphi'\mathcal{S}.$$

Sei β eine zu $\varphi\mathcal{S}$ und $\varphi'\mathcal{S}$ passende Belegung. Dann ist $\beta\mathcal{S}$ passend für φ.

$$\begin{aligned}
\beta \models \varphi\mathcal{S} &\iff \beta\mathcal{S} \models \varphi && \text{nach vorherigem Lemma} \\
&\iff \beta\mathcal{S} \models \varphi' && \text{da } \varphi \equiv \varphi' \\
&\iff \beta \models \varphi'\mathcal{S} && \text{nach vorherigem Lemma}
\end{aligned}$$

Das schließt den Beweis des Substitutionslemmas ab. □

Das Substitutionslemma (formal)

Lemma 2.27. Sei $\mathcal{S}$ eine Substitution und seien $\varphi, \varphi' \in \mathrm{AL}$ Formeln. Dann gilt

$$\varphi \equiv \varphi' \quad \Rightarrow \quad \varphi\mathcal{S} \equiv \varphi'\mathcal{S}.$$

Beweis. Der Beweis wird per Induktion über den Formelaufbau geführt.

Wir führen strukturelle Induktion zunächst allgemein ein und beweisen dann das Lemma.

Notation

Notation 2.28.

1. Sei $\varphi \in \mathsf{AL}$ eine Formel.

   Wenn $X_1, \ldots, X_n$ Variablen in φ asind und $\psi_1, \ldots, \psi_n \in \mathsf{AL}$, schreiben wir

   $$\varphi[X_1/\psi_1, \ldots, X_n/\psi_n]$$

   für die Formel $\varphi\mathcal{S}$, wobei $\mathcal{S}$ die wie folgt definierte Substitution ist

   $$\mathsf{Dom}(\mathcal{S}) := \{X_1, \ldots, X_n\} \text{ und } \mathcal{S}(X_i) := \psi_i.$$

2. Wir schreiben $\varphi(X_1, \ldots, X_n) \in \mathsf{AL}$ um anzudeuten, dass $\varphi \in \mathsf{AL}$ und $\mathsf{var}(\varphi) := \{X_1, \ldots, X_n\}$.

Beispiel

Substitutionslemma. Sei $\mathcal{S}$ eine Substitution und seien $\varphi, \varphi' \in \mathsf{AL}$ Formeln. Dann gilt

$$\varphi \equiv \varphi' \quad \Rightarrow \quad \varphi\mathcal{S} \equiv \varphi'\mathcal{S}.$$

Äquivalenzen. Wir wissen bereits, dass

$$(X \to Y) \equiv \neg X \vee Y \qquad \text{und} \qquad (X \leftrightarrow Y) \equiv (X \to Y) \wedge (Y \to X)$$

Korollar. Für alle Formeln $\varphi, \psi \in \mathsf{AL}$:

$$\begin{aligned} (\varphi \to \psi) &\equiv \neg\varphi \vee \psi \\ (\varphi \leftrightarrow \psi) &\equiv (\varphi \to \psi) \wedge (\psi \to \varphi) \end{aligned}$$

Ersetzungen. Wir würden nun gerne weiter folgern, dass

$$(\varphi \leftrightarrow \psi) \equiv (\varphi \to \psi) \wedge (\psi \to \varphi) \equiv (\neg\varphi \vee \psi) \wedge (\neg\psi \vee \varphi)$$

Die Substitution erlaubt diese Folgerung aber nicht.

Das Ersetzungslemma

Lemma 2.29. Sei $\varphi \in \mathrm{AL}$ eine Formel und ψ eine Unterformel von φ.

Sei φ' eine Formel, die man aus φ erhält, indem man ein Vorkommen der Unterformel ψ durch eine äquivalente Formel $\psi' \equiv \psi$ ersetzt.

Dann gilt $\varphi \equiv \varphi'$.

Bemerkung. Da Unterformeln mehrfach vorkommen können, ist φ' nicht eindeutig.

Mit Hilfe des Ersetzungslemmas können wir nun folgendes beweisen.

Korolar 2.30. Für alle Formeln φ, ψ:

$$(\varphi \leftrightarrow \psi) \quad \equiv \quad (\varphi \rightarrow \psi) \wedge (\psi \rightarrow \varphi) \quad \equiv \quad (\neg\varphi \vee \psi) \wedge (\neg\psi \vee \varphi)$$

Beweis des Ersetzungslemmas

Der Beweis des Ersetzungslemmas geht per struktureller Induktion.

Induktionsanfang. Ist φ atomar, so gilt offenbar $\psi = \varphi$ und somit $\varphi' = \psi'$. Nach Voraussetzung gilt also $\varphi \equiv \varphi'$.

Induktionsschritt. Angenommen, $\varphi := \neg\vartheta$.

- Ist $\psi = \varphi$, so ist nichts zu zeigen.
- Sonst ist ψ eine Teilformel von ϑ und $\varphi' = \vartheta'$, wobei ϑ' die Formel ist, die aus ϑ entsteht, indem ψ durch ψ' ersetzt wird. Nach IV gilt $\vartheta \equiv \vartheta'$.

  Nun gilt also für alle passenden Belegungen β:

$$[\![\varphi]\!]^\beta = 1 - [\![\vartheta]\!]^\beta = 1 - [\![\vartheta']\!]^\beta = [\![\varphi']\!]^\beta$$

  und somit $\varphi \equiv \varphi'$.

Schließlich bleibt noch der Fall $\varphi := (\varphi_1 \wedge \varphi_2)$. Die anderen Fälle sind analog.

Beweis des Ersetzungslemmas

Angenommen, $\varphi := (\varphi_1 \wedge \varphi_2)$.

Wiederum, falls $\psi = \varphi$, so ist nichts zu zeigen.

Sonst können wir o.B.d.A. annehmen, dass ψ eine Unterformel von φ_1 ist. Sei φ_1' die Formel, die aus φ_1 durch Ersetzen von ψ durch ψ' entsteht.

Nach IV gilt also $\varphi_1 \equiv \varphi_1'$ und somit, mit der gleichen Argumentation wie vorher, $\varphi \equiv \varphi'$. $\square$

Nützliche Äquivalenzen

Theorem 2.31. Für alle $\psi, \varphi, \vartheta \in \mathrm{AL}$:

1. $\neg\neg\varphi \equiv \varphi$ (Elimination doppelter Negation)
2. $\varphi \to \psi \equiv \neg\varphi \vee \psi$ (Elim. der Implikation)
3. $\varphi \leftrightarrow \psi \equiv (\varphi \to \psi) \wedge (\psi \to \varphi)$ (Elim. der Biimplikation)
4. $\begin{array}{rcl} \neg(\psi \wedge \varphi) & \equiv & \neg\psi \vee \neg\varphi \\ \neg(\psi \vee \varphi) & \equiv & \neg\psi \wedge \neg\varphi \end{array}$ (de Morgansche Regeln)
5. $\begin{array}{rcl} \psi \wedge (\varphi \vee \vartheta) & \equiv & (\psi \wedge \varphi) \vee (\psi \wedge \vartheta) \\ \psi \vee (\varphi \wedge \vartheta) & \equiv & (\psi \vee \varphi) \wedge (\psi \vee \vartheta) \end{array}$ (Distributivität)
6. $\psi \wedge (\psi \vee \varphi) \equiv \psi \vee (\psi \wedge \varphi) \equiv \psi$ (Absorbtionsgesetz)
7. $\begin{array}{rcl} \psi \wedge \varphi & \equiv & \varphi \wedge \psi \\ \psi \vee \varphi & \equiv & \varphi \vee \psi \end{array}$ (Kommutativität von $\wedge$ und $\vee$)
8. $\begin{array}{rcl} \psi \wedge (\varphi \wedge \vartheta) & \equiv & (\psi \wedge \varphi) \wedge \vartheta \\ \psi \vee (\varphi \vee \vartheta) & \equiv & (\psi \vee \varphi) \vee \vartheta \end{array}$ (Assoziativität von $\wedge$ und $\vee$)

Beweis des Theorems

Wir beweisen hier exemplarisch einige der Äquivalenzen. Die restlichen sind zur Übung empfohlen.

de Morgansche Regel $\neg(\psi \wedge \varphi) \equiv \neg\psi \vee \neg\varphi$. Sei β eine passende Belegung.

Dann gilt

$[\![\neg(\psi \wedge \varphi)]\!]^\beta = 1$	gdw.	wenn $[\![(\psi \wedge \varphi)]\!]^\beta = 0$.
	gdw.	mindestens eins von $[\![\psi]\!]^\beta, [\![\varphi]\!]^\beta$ gleich 0
	gdw.	mindestens eins von $[\![\neg\psi]\!]^\beta, [\![\neg\varphi]\!]^\beta$ gleich 1.
	gdw.	$[\![(\neg\psi \vee \neg\varphi)]\!]^\beta = 1$.

Beweis des Theorems

Distributivität $\psi \vee (\varphi \wedge \vartheta) \equiv (\psi \vee \varphi) \wedge (\psi \vee \vartheta)$.

Sei β eine passende Belegung.

- Ang. $[\![\psi \vee (\varphi \wedge \vartheta)]\!]^\beta = 0$. Also $[\![\psi]\!]^\beta = 0$ und $[\![(\varphi \wedge \vartheta)]\!]^\beta = 0$.

  Es folgt also, dass $[\![\varphi]\!]^\beta = 0$ oder $[\![\vartheta]\!]^\beta = 0$.

  O.B.d.A. sei $[\![\varphi]\!]^\beta = 0$. Dann gilt aber $[\![(\psi \vee \varphi)]\!]^\beta = 0$ und somit $[\![(\psi \vee \varphi) \wedge (\psi \vee \vartheta)]\!]^\beta = 0$.

- Sei also $[\![\psi \vee (\varphi \wedge \vartheta)]\!]^\beta = 1$.

  Es gilt also $[\![\psi]\!]^\beta = 1$ oder $[\![(\varphi \wedge \vartheta)]\!]^\beta = 1$.

  Falls $[\![\psi]\!]^\beta = 1$ so folgt $[\![(\psi \vee \varphi)]\!]^\beta = 1$ und $[\![\psi \vee \vartheta]\!]^\beta = 1$ und somit $[\![(\psi \vee \varphi) \wedge (\psi \vee \vartheta)]\!]^\beta = 1$.

  Anderenfalls gilt $[\![\varphi]\!]^\beta = [\![\vartheta]\!]^\beta = 1$. Dann aber $[\![(\psi \vee \varphi) = [\![\psi \vee \vartheta]\!]^\beta = 1]\!]^\beta$ und somit $[\![(\psi \vee \varphi) \wedge (\psi \vee \vartheta)]\!]^\beta = 1$.

□

Große Disjunktionen und Konjunktionen

Die folgende Notation ist oft nützlich:

$\bigwedge_{i=1}^{n} \varphi_i$ als Abkürzung für $(\varphi_1 \wedge \varphi_2 \wedge \cdots \wedge \varphi_n)$

$\bigvee_{i=1}^{n} \psi_i$ als Abkürzung für $(\varphi_1 \vee \varphi_2 \vee \cdots \vee \varphi_n)$

Beispiel.

$\bigwedge_{i=1}^{999} X_i \to Y$: wenn alle 999 X_i wahr sind, so muss auch Y wahr sein

Bemerkung. Wegen der Assoziativitätsregeln

$$\begin{aligned} \psi \wedge (\varphi \wedge \vartheta) &\equiv (\psi \wedge \varphi) \wedge \vartheta \\ \psi \vee (\varphi \vee \vartheta) &\equiv (\psi \vee \varphi) \vee \vartheta \end{aligned}$$

ändert die Klammerung den Wahrheitswert nicht.

Reduzierte Formeln

Das vorherige Theorem liefert die folgenden Äquivalenzen.

$$
\begin{aligned}
(\varphi \to \psi) &\equiv \neg\varphi \vee \psi \\
(\varphi \leftrightarrow \psi) &\equiv (\varphi \to \psi) \wedge (\psi \to \varphi) \equiv (\neg\varphi \vee \psi) \wedge (\neg\psi \vee \varphi) \\
(\varphi \wedge \psi) &\equiv \neg\big(\neg\varphi \vee \neg\psi\big)
\end{aligned}
$$

Mit Hilfe des Ersetzungslemmas erhalten wir daher folgende Aussage.

Korollar 2.32. Jede aussagenlogische Formel ist äquivalent zu einer Formel ohne $\wedge, \to, \leftrightarrow$, d.h. in der nur $\top$, $\bot$, Variablen und $\vee$ und $\neg$ vorkommen.

Wir nennen solche Formeln reduziert.

Reduzierte Formeln sind nützlich, um die Zahl der Fälle in strukturellen Induktionen zu begrenzen.

Normalformen

Negationsnormalform

Definition 2.33. Eine Formel $\varphi \in \mathrm{AL}$ ist in Negationsnormalform (NNF), wenn die Symbole $\rightarrow$ und $\leftrightarrow$ nicht vorkommen und Negation nur vor Variablen auftritt.

Beispiel. $\varphi := (\neg X \vee Y) \wedge \neg Z$

Theorem 2.34. Jede Formel $\varphi \in \mathrm{AL}$ ist äquivalent zu einer Formel $\varphi^* \in \mathrm{AL}$ in Negationsnormalform.

Beweis. Wir wissen bereits, dass φ zu einer Formel φ' ohne $\rightarrow, \leftrightarrow$ äquivalent ist.

O.B.d.A. nehmen wir daher an, dass $\rightarrow, \leftrightarrow$ nicht in φ vorkommen.

Durch Anwendung der Äquivalenzen 1. und 4. können wir die Formel in Negationsnormalform umwandeln.

Ein Algorithmus für die NNF

Eingabe. Eine Formel $\varphi \in \mathrm{AL}$ ohne $\rightarrow, \leftrightarrow$.

Ausgabe. Eine Formel $\varphi^* \equiv \varphi$ in NNF

Algorithmus. Wiederhole die folgenden Schritte.

Wenn φ in NNF ist, gib $\varphi^* := \varphi$ aus.

Wenn φ eine Unterformel ψ der Form $\neg\neg\psi_1$ enthält, dann ersetze ψ durch ψ_1 um φ' zu erhalten.

Wenn φ eine Unterformel ψ der Form $\neg(\psi_1 \wedge \psi_2)$ enthält, dann ersetze ψ durch $(\neg\psi_1 \vee \neg\psi_2)$ um φ' zu erhalten.

Wenn φ eine Unterformel ψ der Form $\neg(\psi_1 \vee \psi_2)$ enthält dann ersetze ψ durch $(\neg\psi_1 \wedge \neg\psi_2)$ um φ' zu erhalten.

Setze $\varphi := \varphi'$.

Lemma 2.35. Der Algorithmus terminiert auf jeder gültigen Eingabe $\varphi \in \mathrm{AL}$ und konstruiert eine Formel φ^* in NNF, so dass $\varphi \equiv \varphi^*$.

Beispiel

Sei $\varphi := \neg(\neg(X \vee Y) \wedge (\neg(X \wedge Y) \vee Z))$. Dann erzeugt der Algorithmus folgende Zwischenformeln:

$$
\begin{aligned}
& \neg(\neg(X \vee Y) \wedge (\neg(X \wedge Y) \vee Z)) \\
\equiv\ & (\neg\neg(X \vee Y) \vee \neg(\neg(X \wedge Y) \vee Z)) \\
\equiv\ & ((X \vee Y) \vee \neg(\neg(X \wedge Y) \vee Z)) \\
\equiv\ & ((X \vee Y) \vee \neg((\neg X \vee \neg Y) \vee Z)) \\
\equiv\ & ((X \vee Y) \vee (\neg(\neg X \vee \neg Y) \wedge \neg Z)) \\
\equiv\ & ((X \vee Y) \vee ((\neg\neg X \wedge \neg\neg Y) \wedge \neg Z)) \\
\equiv\ & ((X \vee Y) \vee ((X \wedge \neg\neg Y) \wedge \neg Z)) \\
\equiv\ & ((X \vee Y) \vee ((X \wedge Y) \wedge \neg Z))
\end{aligned}
$$

Beweis des Lemmas

Äquivalenz. Der Algorithmus ersetzt in jedem Schritt eine Unterformel ψ von φ durch eine äquivalente Formel. Nach dem Ersetzungslemma sind daher φ und φ' äquivalent.

Terminierung. Wir definieren eine Funktion

$$h : \mathsf{AL} \to \mathbb{N},$$

die die Tiefe einer Formel angibt, wie folgt:

- Ist ψ atomar, so gilt $h(\psi) := 0$.
- Ist $\psi := \neg\psi'$, so gilt $h(\psi) := 1 + f(\psi')$.
- Ist $\psi := (\psi_1 \vee \psi_2)$ oder $\psi := (\psi_1 \wedge \psi_2)$, so gilt $h(\psi) := 1 + \max\{f(\psi_1), f(\psi_2)\}$.

Die Tiefe einer Formel ist also die Höhe des Syntaxbaums der Formel.

Beobachtung. Eine Formel ist in NNF genau dann, wenn jede Unterformel der Form $\neg\psi'$ die Tiefe 1 hat.

Beweis des Lemmas

Sei jetzt $f(\varphi) := \sum\{3^{h(\psi)} : \psi = \neg\psi' \in \mathrm{sub}(\varphi)\}$

Behauptung. Sei φ eine Formel und φ' die Formel, die aus φ in einem Schritt des Algorithmus' entsteht. Dann gilt $f(\varphi') < f(\varphi)$.

Beweis. Angenommen, $\psi := \neg\neg\psi_1$. Wir ersetzen also ψ durch ψ_1. Dadurch erhöht sich die Tiefe der restlichen Negationsformeln nicht. Die Zahl solcher Formeln reduziert sich um 2 und somit gilt $f(\varphi') < f(\varphi)$.

Angenommen, $\psi := \neg(\psi_1 \vee \psi_2)$ oder $\psi := \neg(\psi_1 \wedge \psi_2)$. Dann bleibt die Tiefe aller Negationsformeln außer ψ gleich. In der Summierung wird also $3^{h(\psi)}$ durch $3^{h(\neg\psi_1)} + 3^{h(\neg\psi_2)} = 2 \cdot 3^{h(\psi)-1} < 3^{h(\psi)}$ ersetzt. Also gilt $f(\varphi') < f(\varphi)$. □

Hinweis. Ein Beispiel wird in den Tutorien besprochen.

Normalformen

Definition 2.36. Ein Literal L ist eine Aussagenvariable $X \in \text{AVAR}$ oder deren Negation $\neg X$.
Wir schreiben $\overline{L}$ für das Komplementliteral definiert als

$$\overline{L} := \begin{cases} \neg X & \text{if } L = X \\ X & \text{if } L = \neg X. \end{cases}$$

Definition 2.37. Eine Formel $\varphi \in \text{AL}$ ist in disjunktiver Normalform (DNF), wenn sie folgende Gestalt hat:

$$\bigvee_{i=1}^{n} \left(\bigwedge_{j=1}^{n_i} L_{i,j} \right)$$

φ ist in konjunktiver Normalform (KNF), wenn sie folgende Gestalt hat:

$$\bigwedge_{i=1}^{n} \left(\bigvee_{j=1}^{n_i} L_{i,j} \right)$$

Normalformen

Theorem 2.38.

1. Jede Formel $\varphi \in AL$ ist äquivalent zu einer Formel in disjunktiver Normalform.
2. Jede Formel $\varphi \in AL$ ist äquivalent zu einer Formel in konjunktiver Normalform.

Beweis. Das Theorem kann ähnlich wie der Satz über die Negationsnormalform bewiesen werden (Übung).

Wir verwenden hier einen anderen Ansatz über Boolesche Funktionen.

Einschub: Boolesche Funktionen

Boolesche Funktionen

Definition 2.39. Eine (n-stellige) Boolesche Funktion, nach Georg Boole benannt, ist eine Funktion

$$f : \{0,1\}^n \to \{0,1\}.$$

Wir definieren $\mathbb{B}^n$ als Menge aller n-stelligen Booleschen Funktionen.

Beispiel. Definiere

$$f(X_1,\ldots,X_n) := \begin{cases} 1 & \text{wenn die Mehrheit der } X_i \text{ gleich 1 sind} \\ 0 & \text{sonst} \end{cases}$$

Proposition 2.40. Jede Formel $\varphi(X_1,\ldots,X_n) \in \mathrm{AL}$ definiert eine Boolesche Funktion $f_\varphi := f(\varphi)$ mit

$$\begin{aligned} f_\varphi &: \{0,1\}^n \to \{0,1\} \\ f_\varphi(v_1,\ldots,v_n) &:= [\![\varphi]\!]^\beta \end{aligned}$$

wobei $\beta(X_i) := v_i$, für alle $1 \leqslant i \leqslant n$.

Boolesche Funktionen

Umgekehrt kann jede Boolesche Funktion durch eine Formel definiert werden.

Theorem 2.41. Zu jeder Booleschen Funktion $f : \{0,1\}^n \to \{0,1\}$ gibt es eine Formel $\varphi(X_1, \ldots, X_n)$, so dass $f(\varphi) = f$.

Beweis. Für jede Sequenz $\overline{v} := (v_1, \ldots, v_n) \in \{0,1\}^n$ definieren wir

$$\varphi_{\overline{v}} := \left(\bigwedge_{v_i=1} X_i\right) \wedge \left(\bigwedge_{v_i=0} \neg X_i\right)$$

Offensichtlich gilt für jede Belegung β, wenn $\beta \models \varphi_{\overline{v}}$ dann gilt für alle $1 \leqslant i \leqslant n$:

$$\beta(X_i) = 1 \iff v_i = 1.$$

Wir definieren nun die Funktion f durch die Formel

$$\varphi_f(X_1, \ldots, X_n) := \bigvee_{\substack{\overline{v} \in \{0,1\}^n \\ f(\overline{v})=1}} \varphi_{\overline{v}}.$$

Boolesche Funktionen

Wir müssen nun zeigen, dass für alle $\overline{v} := (v_1, \ldots, v_n)$:

$$f(\overline{v}) = 1 \iff [\![\varphi_f]\!]^\beta$$

wobei $\beta(X_i) := v_i$, für alle $1 \leqslant i \leqslant n$.

1. Wenn $f(\overline{v}) = 1$, dann $\beta \models \varphi_{\overline{v}}$ und $\varphi_{\overline{v}}$ ist Teil der Disjunktion in φ_f.

   Also $\beta \models \varphi_f$.

2. Umgekehrt, wenn $\beta \models \varphi_f$, dann muss es ein Disjunktionsglied $\varphi_{\overline{w}}$ geben, so dass $\beta \models \varphi_{\overline{w}}$.

   Nach Konstruktion von φ_f gilt $f(\overline{w}) = 1$.

   Aber $\beta \models \varphi_{\overline{w}}$ genau dann, wenn $w_i = \beta(X_i) = v_i$ für alle $1 \leqslant i \leqslant n$.

   Also $f(\overline{v}) = 1$.

Das schließt den Beweis ab. □

Beispiel

Beispiel. Definiere

$$f(X_1, X_2, X_3) := \begin{cases} 1 & \text{falls die Mehrheit der } X_i \text{ gleich 1 ist} \\ 0 & \text{sonst} \end{cases}$$

$f(v_1, v_2, v_3) = 1$ wenn mindestens zwei der v_i gleich 1 sind.

Wir erhalten also die folgende Formel

$$\begin{array}{rcll} \varphi_f(X_1, X_2, X_3) & := & (\neg X_1 \wedge X_2 \wedge X_3) & (=: \varphi_{0,1,1}) \\ & \vee & (X_1 \wedge \neg X_2 \wedge X_3) & (=: \varphi_{1,0,1}) \\ & \vee & (X_1 \wedge X_2 \wedge \neg X_3) & (=: \varphi_{1,1,0}) \\ & \vee & (X_1 \wedge X_2 \wedge X_3) & (=: \varphi_{1,1,1}) \end{array}$$

Formeln vs. Booleschen Funktionen

Wir haben folgenden Zusammenhang zwischen aussagenlogischen Formeln und Booleschen Funktionen gezeigt:

1. Jede Formel $\varphi(X_1, \ldots, X_n) \in \mathrm{AL}$ definiert eine Boolesche Funktion $f_\varphi := f(\varphi)$.
2. Zu jeder Booleschen Funktion $f : \{0,1\}^n \to \{0,1\}$ gibt es eine Formel $\varphi(X_1, \ldots, X_n)$, so dass $f(\varphi) = f$.

D. h. es besteht ein eins-zu-eins Zusammenhang zwischen Formeln und Booleschen Funktionen.

Korollar 2.42. Für alle $n \geqslant 0$ existieren genau 2^{2^n} paarweise nicht-äquivalente aussagenlogische Formeln in den Variablen $X_1, \ldots, X_n$.

Beweis. Es gibt 2^n verschiedene Belegungen der Variablen $X_1, \ldots, X_n$. Also existieren 2^{2^n} verschiedene n-stellige Boolesche Funktionen und somit ebensoviele aussagenlogische Formeln in den Variablen $X_1, \ldots, X_n$.

Zurück zu Normalformen

Normalformen

Theorem 2.43.

1. Jede Formel $\varphi \in \mathrm{AL}$ ist äquivalent zu einer Formel in disjunktiver Normalform.
2. Jede Formel $\varphi \in \mathrm{AL}$ ist äquivalent zu einer Formeln in konjunktiver Normalform.

Beweis. Teil 1 folgt sofort aus dem Beweis des vorherigen Theorems, da die dort konstruierten Formeln in disjunktiver Normalform sind.

Zu Teil 2: Sei $\varphi \in \mathrm{AL}$.

Nach Teil 1 ist $\neg\varphi$ äquivalent zu einer Formel $\psi := \bigvee_{i=1}^{n} \bigwedge_{j=1}^{n_i} L_{i,j}$ in DNF.

Mit Hilfe der de Morganschen Gesetze erhält man

$$\varphi \equiv \neg \bigvee_{i=1}^{n} \bigwedge_{j=1}^{n_i} L_{i,j} \quad \equiv \quad \bigwedge_{i=1}^{n} \bigvee_{j=1}^{n_i} \overline{L_{i,j}}.$$

□

Normalformen

Bemerkung.

- Formeln in disjunktiver Normalform können sehr effizient auf Erfüllbarkeit getestet werden.
- Allerdings gibt es Formeln $\varphi_n \in \mathrm{AL}$, für alle $n \in \mathbb{N}$, so dass die Länge der kürzesten zu φ_n äquivalenten Formeln in DNF exponentiell in der Länge von φ_n sind.
- Es gibt also im Allgemeinen keinen effizienten Weg um aussagenlogische Formeln in disjunktive oder konjunktive Normalform umzuwandeln.
- Jedoch kann zu jeder Formel $\varphi \in \mathrm{AL}$ in Polynomialzeit eine Formel $\psi \in \mathrm{AL}$ in konjunktiver Normalform konstruiert werden, so dass

  φ genau dann erfüllbar ist, wenn ψ erfüllbar ist.

  Dies wird in praktischen Anwendungen benutzt, da die meisten aktuellen SAT-Löser Formeln in KNF als Eingabe erwarten.

Funktional vollständige Mengen von Verknüpfungen

Funktionale Vollständigkeit

Wie wir bereits gesehen haben, können die Verknüpfungen $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$ auf natürliche Art als Boolesche Funktionen aufgefasst werden.

So entspricht z.B. $\wedge$ der Funktion $f_\wedge(X_1, X_2) \in \mathbb{B}^2$ mit $f_\wedge(X_1, X_2) = 1$ genau dann, wenn $X_1 = X_2 = 1$.

Umgekehrt kann man jede Boolesche Funktion $f \in \mathbb{B}^n$ als n-stellige Verknüpfung auffassen:

Aus Formeln $\varphi_1, \ldots, \varphi_n$ können wir die Formel $f(\varphi_1, \ldots, \varphi_n)$ bilden.

Beispiel. Sei $\oplus \in \mathbb{B}^2$ (XOR) die Boolesche Funktion definiert durch

$$\oplus(X_1, X_2) = 1 \text{ genau dann, wenn } X_1 \neq X_2.$$

Wir können dann Formeln $(\oplus(X, Y) \wedge \oplus(Y, Z))$ verwenden.

Funktionale Vollständigkeit

Wir können also in der Aussagenlogik auch andere Verknüpfungen zulassen und zum Beispiel die Logik auf den Verknüpfungen $\wedge, \neg, \oplus$ aufbauen.

Je nach verwendeten Verknüpfungen erhalten wir „äquivalente" Logiken oder aber Logiken, in denen nicht mehr alle Booleschen Funktionen definiert werden können.

Definition 2.44.

- Eine Menge $M \subseteq \mathbb{B}$ ist funktional vollständig, wenn sich daraus jede Boolesche Funktion im Sinne des vorherigen Satzes konstruieren läßt.
- Eine Menge V von Verknüpfungen ist funktional vollständig, wenn die entsprechende Menge Boolescher Funktionen funktional vollständig ist.

Hierbei bezeichnet $\mathbb{B} := \bigcup_n \mathbb{B}^n$ die Menge aller Booleschen Funktionen.

Funktionale Vollständigkeit

Beispiel. Wir wissen bereits, dass jede aussagenlogische Formel, und somit jede Boolesche Funktion, zu einer Formel in disjunktiver Normalform äquivalent ist.

Da in DNF nur die Verknüpfungen $\wedge, \vee, \neg$ vorkommen, ist also die Menge $\wedge, \vee, \neg$ funktional vollständig.

Funktionale Vollständigkeit

Beispiel Die Menge $\neg, \vee$ ist funktional vollständig.

Begründung.

Um zu zeigen, dass eine Menge V von Verknüpfungen funktional vollständig ist, reicht es zu zeigen, dass die Verknüpfungen einer funktional vollständigen Menge V' von Verknüpfungen aus V ausgedrückt werden können.

Wir wissen bereits, dass $(\varphi \wedge \psi) \equiv \neg(\neg\varphi \vee \neg\psi)$. Also ist $\neg, \vee$ funktional vollständig, da $\wedge, \vee, \neg$ funktional vollständig ist. $\square$

Bemerkung.

- Ebenso zeigt man, dass $\neg, \wedge$ funktional vollständig ist.
- Die Menge $\wedge, \vee, \rightarrow$ ist nicht funktional vollständig.

  Zum Beispiel kann keine zu $\neg X$ äquivalente Formel gebildet werden (Übung).

Semantische Folgerung

Semantische Folgerung

Erinnern wir uns an das Beispiel vom Anfang der Vorlesung:

Voraussetzungen.

- Wenn der Zug zu spät ist und keine Taxis am Bahnhof stehen, kommt Peter zu spät zu seiner Verabredung.
- Peter kam nicht zu spät zu seiner Verabredung.
- Der Zug hatte Verspätung.

Folgerung. Also standen Taxis am Bahnhof.

Aussagenvariablen:

T: Zug zu spät C: Taxis am Bahnhof L: Peter kam zu spät

Die Argumentation kann wie folgt zusammengefasst werden:

Aus den Voraussetzungen: $\{(T \wedge \neg C) \rightarrow L, \quad \neg L, \quad T\}$

folgern wir Folgerung: C

Semantische Folgerung

Frage. Wie können wir solche logischen Schlüsse formalisieren?

- Gegeben eine Menge von Voraussetzungen:

  $$\{ \begin{array}{l} \text{“Zug zu spät und keine Taxis impliziert Peter zu spät",} \\ \text{“Peter nicht zu spät,} \quad \text{“Zug zu spät”} \end{array} \}$$

  können wir daraus “es gab Taxis am Bahnhof” schließen?

- Gibt es allgemeine Methoden, solche Folgerungen aus den Voraussetzungen zu ziehen? Methoden, mit denen
  - nur logisch korrekte Folgerungen abgeleitet werden können, die aber
  - allgemein genug sind, damit alle logisch korrekten Folgerungen abgeleitet werden können?

Damit zusammenhängend stellen sich algorithmische Fragen.

Frage. Wie können wir solche Folgerungen berechnen?

- Die Wahrheitstafelmethode kann für endliche Voraussetzungsmengen benutzt werden.
- Allerdings ist die Methode sehr ineffizient, da die Wahrheitstafeln sehr groß werden können.

Semantische Folgerung

Wir führen zunächst den Folgerungsbegriff zwischen Formelmengen und Formeln ein, einen der wichtigsten Begriffe der Logik überhaupt, nicht nur für die Aussagenlogik.

Definition 2.45. Sei $\Phi \subseteq \mathrm{AL}$ eine Formelmenge und $\psi \in \mathrm{AL}$ eine Formel.

ψ folgt aus Φ, wenn jede zu $\Phi \cup \{\psi\}$ passende Belegung β, die Φ erfüllt, auch ψ erfüllt.

Wir schreiben $\Phi \models \psi$.

Falls $\Phi := \{\varphi\}$ nur eine Formel enthält, schreiben wir nur $\varphi \models \psi$.

Bemerkung. Wir verwenden das Symbol $\models$ sowohl für die Modellbeziehung $\beta \models \psi$ als auch für die semantische Folgerung $\Phi \models \psi$.

Eigenschaften der Folgerungsbeziehung

Das folgende Lemma listet einige einfache Eigenschaften der Folgerungsbeziehung auf, die sofort aus der Definition folgen.

Lemma 2.46. Sei $\Phi \subseteq \mathsf{AL}$ und $\psi, \psi' \in \mathsf{AL}$.

1. $\psi \equiv \psi'$ genau dann, wenn $\psi \models \psi'$ und $\psi' \models \psi$.
2. $\Phi \models \psi$ genau dann, wenn $\Phi \cup \{\neg\psi\}$ unerfüllbar ist.
3. Sei $\Phi_0 \subseteq \Phi$. Wenn $\Phi_0 \models \psi$, dann auch auch $\Phi \models \psi$.

Semantische Folgerung

Aussagenvariablen:

T: Zug zu spät C: Taxis am Bahnhof L: Peter kam zu spät

Die Argumentation kann wie folgt zusammengefasst werden:

Aus den Voraussetzungen: $\{(T \wedge \neg C) \rightarrow L, \quad \neg L, \quad T\}$

folgern wir Folgerung: C

Semantische Folgerung. Das bedeutet: Wir wollen zeigen, dass

$$\{(T \wedge \neg C) \rightarrow L, \quad \neg L, \quad T\} \quad \models \quad C$$

Semantische Folgerung

Wir werden in der Vorlesung zwei Methoden kennen lernen, mit denen semantische Folgerungen automatisch und elegant überprüft werden können.

- Aussagenlogische Resolution (jetzt)
- Der aussagenlogische Sequenzenkalkül (später in der VL)

Zunächst jedoch werden wir einen Satz beweisen, der uns in bestimmten Situationen auch die Behandlung unendlicher Formelmengen erlaubt bzw. vereinfacht.

2.6. Der Kompaktheitssatz der Aussagenlogik

Der Kompaktheitssatz

In bestimmten Anwendungen der Aussagenlogik treten unendliche Formelmengen auf, die auf Erfüllbarkeit untersucht werden sollen.

Wir werden als nächstes einen Satz beweisen, der uns diese Aufgabe wesentlich vereinfachen wird, da er es erlaubt, Erfüllbarkeit unendlicher Formelmengen auf Erfüllbarkeit endlicher Formelmengen zu reduzieren.

Der Kompaktheitssatz der Aussagenlogik

Definition 2.47.

Eine Menge Φ aussagenlogischer Formeln ist erfüllbar, wenn es eine Belegung β gibt, die zu allen $\varphi \in \Phi$ passt und alle $\varphi \in \Phi$ erfüllt.

Wir schreiben wiederum $\beta \models \Phi$.

Theorem (Kompaktheits- oder Endlichkeitssatz)

Sei $\Phi \subseteq \mathrm{AL}$ eine Formelmenge und $\psi \in \mathrm{AL}$ eine Formel.

1. Φ ist genau dann erfüllbar, wenn jede endliche Teilmenge $\Phi' \subseteq \Phi$ erfüllbar ist.
2. $\Phi \models \psi$ genau dann, wenn eine endliche Teilmenge $\Phi_0 \subseteq \Phi$ existiert, so dass $\Phi_0 \models \psi$.

Bemerkung. Der Satz kann auch für überabzählbare Variablen und somit überabzählbare Formelmengen bewiesen werden.

Beweis des Satzes

Wir werden zunächst den ersten Teil des Satzes beweisen, d.h.

> *Eine Menge $\Phi \subseteq \mathrm{AL}$ ist genau dann erfüllbar, wenn jede endliche Teilmenge $\Phi' \subseteq \Phi$ erfüllbar ist.*

Vorüberlegungen.

Offenbar ist das Lemma trivial, wenn Φ bereits endlich ist.

Sei Φ also eine unendliche Menge aussagenlogischer Formeln.

Ohne Beschränkung der Allgemeinheit nehmen wir an, dass es keine zwei verschiedenen Formeln $\psi, \psi' \in \Phi$ gibt, so dass $\psi \equiv \psi'$.

Denn, angenommen, es gäbe solche Formeln. Dann ist Φ genau dann erfüllbar, wenn $\Phi \setminus \{\psi'\}$ erfüllbar ist.

Es reicht also, den Beweis für Formelmengen zu zeigen, in denen alle Formeln paarweise nicht äquivalent sind.

Beweis des Satzes

Hinrichtung.

Zum Beweis der Hinrichtung sei Φ erfüllbar. Also existiert eine Belegung β, die jede Formel in Φ erfüllt. Also ist auch jede endliche Teilmenge von Φ erfüllbar, z. B. durch die Belegung β.

Rückrichtung.

Angenommen, alle endlichen Teilmengen $\Phi' \subseteq \Phi$ sind erfüllbar.

Seien $X_1, X_2, \ldots$ die in Formeln in Φ vorkommenden Aussagenvariablen.

Für $n \geqslant 0$ sei $\Phi_n \subseteq \Phi$ die Menge aller Formeln aus Φ in denen nur die Variablen $X_1, \ldots, X_n$ vorkommen (es müssen aber nicht alle vorkommen).

Es gilt also $\Phi_0 \subseteq \Phi_1 \subseteq \ldots \subseteq \Phi$.

Beweis des Satzes

Wir haben bereits bewiesen, dass es höchstens 2^{2^n} paarweise nicht-äquivalente Formeln in n Variablen gibt. Da Φ keine paarweise äquivalenten Formeln enthält, umfasst jedes Φ_n höchstens 2^{2^n} Formeln und ist damit endlich.

Nach Voraussetzung gibt es also für jedes $n \geqslant 0$ eine Belegung β_n, so dass $\beta_n \models \Phi_n$ und somit auch $\beta_n \models \Phi_i$ für alle $i \leqslant n$.

Sei $I_0 := \{\beta_n : n \geqslant 0\}$.

Wir konstruieren induktiv eine Belegung $\alpha : \{X_1, ...\} \to \{0, 1\}$ und Mengen $I_n \subseteq I_0$ wie folgt.

Dabei bewahren wir für alle n stets folgende Eigenschaft $(*)$:

- I_n ist unendlich, $\beta_1, \ldots, \beta_{n-1} \notin I_n$ und
- für alle $\beta, \beta' \in I_n$ und $j \leqslant n$ gilt $\beta(X_j) = \beta'(X_j) = \alpha(X_j)$.

Beweis des Satzes

Induktionsbasis $n = 1$. Da Φ unendlich ist, existiert ein $t \in \{0, 1\}$ so dass $\beta_n(X_1) = t$ für unendlich viele $\beta_n \in I_0$ gilt.

Setze $\alpha(X_1) := t$ und $I_1 := \{\beta \in I_0 : \beta(X_1) = t \text{ und } \beta \neq \beta_1\}$.

Offenbar ist $(*)$ erfüllt.

Induktionsvoraussetzung. Seien I_{n-1}, $\alpha(X_i)$ für alle $i < n$ schon konstruiert so dass $(*)$ gilt.

Induktionsschritt. Da I_{n-1} wegen $(*)$ unendlich ist, gibt es ein $t \in \{0, 1\}$, so das $\beta(X_n) = t$ für unendlich viele $\beta \in I_{n-1}$.

Setze $\alpha(X_n) := t$ und $I_n := \{\beta \in I_{n-1} : \beta(X_n) = t \text{ und } \beta \neq \beta_{n-1}\}$.

Offenbar ist $(*)$ erfüllt.

Beweis des Satzes

Behauptung. $\alpha \models \Phi$.

Sei $\varphi \in \Phi$.

Da φ nur endlich viele Variablen enthält, ist $\varphi \in \Phi_n$ für ein n.

Es gilt also $\beta_i \models \varphi$ für alle $i \geqslant n$, insbesondere also $\beta \models \varphi$ für alle $\beta \in I_n$.

Sei $\beta \in I_n$. So ein β existiert, da wegen $(*)$ $I_n \neq \emptyset$.

Da nach $(*)$ für alle $i \leqslant n$ gilt $\alpha(X_i) = \beta(X_i)$ und $\beta \models \varphi$, folgt also $\alpha \models \varphi$. □

Der Kompaktheitssatz der Aussagenlogik

Theorem (Kompaktheits- oder Endlichkeitssatz)

Sei $\Phi \subseteq \mathrm{AL}$ eine Formelmenge und $\psi \in \mathrm{AL}$ eine Formel.

1. Φ ist genau dann erfüllbar, wenn jede endliche Teilmenge $\Phi' \subseteq \Phi$ erfüllbar ist.
2. $\Phi \models \psi$ genau dann, wenn eine endliche Teilmenge $\Phi_0 \subseteq \Phi$ existiert, so dass $\Phi_0 \models \psi$.

Beweis von Teil 2.

Offenbar gilt $\Phi \models \psi$ genau dann, wenn $\Phi \cup \{\neg\psi\}$ unerfüllbar ist.

Dies ist aber nach Teil 1. genau dann der Fall, wenn bereits eine endliche Teilmenge Φ_0 unerfüllbar ist.

- Ist $\neg\psi \in \Phi_0$, so gilt also $\Phi_0 \setminus \{\neg\psi\} \models \psi$.
- Anderenfalls ist $\Phi_0 \subseteq \Phi$, und da Φ_0 unerfüllbar ist, folgt $\Phi_0 \models \psi$.

Die Umkehrung ist trivial. □

Eine Anwendung

Definition. Ein Graph $G := (V, E)$ besteht aus einer Knotenmenge V und einer Kantenmenge $E \subseteq \{\{u, v\} : u \neq v, u, v \in V\}$.

G ist 3-färbbar, wenn es eine Funktion $c : V \to \{C_1, C_2, C_3\}$ gibt, so dass $c(u) \neq c(v)$ für alle Kanten $\{u, v\} \in E$.

Lemma. Ein Graph ist genau dann 3-färbbar, wenn bereits jeder endliche Untergraph 3-färbbar ist.

Beweis. Übung.

2.7. Aussagenlogische Resolution

Aussagenlogische Resolution

Wir werden in diesem Abschnitt eine Methode kennen lernen, um die Unerfüllbarkeit aussagenlogischer Formeln nachzuweisen.

Da für eine Formelmenge $\Phi \subseteq AL$ und eine Formel ψ gilt:

$$\Phi \models \psi \text{ genau dann, wenn } \Phi \cup \neg\psi \text{ unerfüllbar ist}$$

können mit Hilfe der Resolution auch semantische Folgerungen überprüft werden.

Aussagenlogische Resolution: Beispiel

Wir wollen zeigen, dass die folgende Formel φ unerfüllbar ist.

$$\neg V \wedge (Y \vee Z \vee V) \wedge (\neg X \vee \neg Z) \wedge (X \vee \neg Z) \wedge (\neg Y \vee W) \wedge (\neg W \vee Z)$$

Angenommen, φ wäre erfüllbar. Sei β eine Belegung, so dass $\beta \models \varphi$.

- Offensichtlich gilt, $\beta \models \neg V$
- Aus $\beta \models \neg V$ und $\beta \models Y \vee Z \vee V$ folgt $\beta \models Y \vee Z$
- Aus $\beta \models Y \vee Z$ und $\beta \models \neg Y \vee W$ folgt $\beta \models Z \vee W$
- Aus $\beta \models Z \vee W$ und $\beta \models \neg W \vee Z$ folgt $\beta \models Z$
- Aus $\beta \models \neg X \vee \neg Z$ und $\beta \models X \vee \neg Z$ folgt aber auch $\beta \models \neg Z$
- Offensichtlich ist das ein Widerspruch zu $\beta \models Z$.

Aussagenlogische Resolution

Die aussagenlogische Resolution ist eine Methode um zu zeigen, dass eine Formel in konjunktiver Normalform nicht erfüllbar ist.

Wir haben bereits gezeigt, dass jede Formel $\varphi \in \mathrm{AL}$ zu einer Formel ψ in KNF äquivalent ist.

Also kann die Resolutionsmethode für alle Formeln verwendet werden, indem sie zunächst in KNF umgewandelt werden.

Für praktische Anwendungen der Resolutionsmethode ist folgender Satz nützlich.

Theorem 2.48. Zu jeder Formel φ gibt es eine Formel ψ in KNF, so dass

1. φ ist genau dann erfüllbar, wenn ψ erfüllbar ist.
2. $|\psi| \leqslant c \cdot |\varphi|$ für eine Konstante $c \in \mathbb{N}$ unabhängig von φ.
3. ψ kann aus φ effizient (in Linearzeit) berechnet werden.

Notation

Um das Schreiben von Resolutionsableitungen zu vereinfachen, verwenden wir folgende Notation.

Eine Formel

$$(\neg X \vee \neg Z) \wedge (X \vee \neg Z) \wedge (\neg Y \vee W) \wedge (Y \vee Z \vee V) \wedge \neg V \wedge (\neg W \vee Z)$$

in KNF schreiben wir als Klauselmenge wie folgt:

$$\{\neg X, \neg Z\}, \quad \{X, \neg Z\}, \quad \{\neg Y, W\}, \quad \{Y, Z, V\}, \quad \{\neg V\}, \quad \{\neg W, Z\}$$

D.h., aus jeder Disjunktion $Y \vee Z \vee V$ wird eine als Klausel bezeichnete Menge.

Klauseln

Definition 2.49.

- Eine Klausel ist eine endliche Menge von Literalen.
- Die leere Klausel wird mit $\square$ bezeichnet.
- Mit jeder Formel $\varphi := \bigwedge_{i=1}^{n} \bigvee_{j=1}^{m_i} L_{i,j}$ in KNF assoziieren wir eine endliche Menge $\mathcal{C}(\varphi)$ von Klauseln wie folgt:
  - mit jeder Disjunktion $\bigvee_{j=1}^{m_i} L_{i,j}$ assoziieren wir die Klausel $C_i := \{L_{i,j} : 1 \leqslant j \leqslant m_i\}$
  - wir definieren $\mathcal{C}(\varphi) := \{C_1, \ldots, C_n\}$.
- Umgekehrt, entspricht jeder Menge $\mathcal{C} := \{C_1, \ldots, C_n\}$ von Klauseln

  $$C_i := \{L_{i,j} : 1 \leqslant j \leqslant m_i\}$$

  die Formel $\varphi(\mathcal{C}) := \bigwedge_{i=1}^{n} \bigvee_{j=1}^{m_i} L_{i,j}$.

  Falls $\mathcal{C} := \emptyset$, definieren wir $\varphi(\mathcal{C}) := \top$.

- Der leeren Klausel $\square$ wird die Formel $\bot$ zugeordnet.

Formeln vs. Klauselmengen

Formeln in KNF und Klauselmengen entsprechen sich eins zu eins.

Wir erweitern daher Notation für Formeln auf naheliegende Weise auf Klauselmengen.

- Für eine Belegung β und Klauselmenge $\mathcal{C}$ schreiben wir $\beta \models \mathcal{C}$ für $\beta \models \varphi(\mathcal{C})$
- Wir schreiben $\mathcal{C} \models C$ um zu sagen, dass jede $\mathcal{C}$ erfüllende Belegung auch C erfüllt.
- Eine Klauselmenge $\mathcal{C}$ ist erfüllbar, wenn $\varphi(\mathcal{C})$ erfüllbar ist.
- Wenn $\mathcal{C}$ nur eine Klausel C enthält, schreiben wir einfach nur $\beta \models C$ etc.

Offenbar erfüllt eine Belegung β eine Klauselmenge $\mathcal{C}$, wenn jede Klausel $C \in \mathcal{C}$ ein Literal L enthält, so dass $[\![L]\!]^\beta = 1$.

Insbesondere ist also jede Klauselmenge, die die leere Klausel enthält, unerfüllbar.

Resolution

Notation. Für Literal L ist $\bar{L}$ das duale Literal, d.h. $\bar{X} = \neg X$ und $\overline{\neg X} = X$.

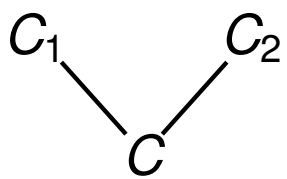
Definition 2.50. Seien C, C_1, C_2 Klauseln.

C ist eine Resolvente von C_1, C_2, wenn es ein Literal L gibt mit

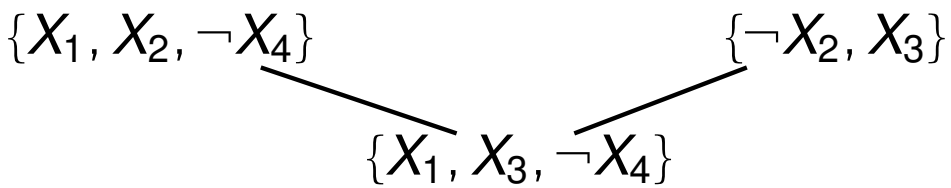
$$L \in C_1 \text{ und } \bar{L} \in C_2 \text{ und } C = (C_1 \setminus \{L\}) \cup (C_2 \setminus \{\bar{L}\}).$$

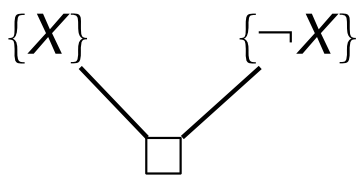
Wir sagen, dass C_1 und C_2 resolviert werden und schreiben $Res(C_1, C_2)$ für die Menge der Resolventen von C_1 und C_2.

Wir werden das oft wie folgt graphisch darstellen



Beispiel.





Resolution

Anmerkung. Zwei Klauseln können mehr als eine Resolvente haben, d.h. $Res(C_1, C_2)$ kann mehr als ein Element enthalten.

Die Klauseln $\{X, Y, Z\}$ und $\{\neg X, \neg Y, W\}$ haben als Resolventen $\{Y, \neg Y, Z, W\}$ und $\{X, \neg X, Z, W\}$.

Dies ist aber ein degenerierter Fall, der im weiteren keine Rolle spielen wird. Insbesondere ist die Resolvente immer allgemeingültig, da sie ein Literal und sein duales Literal enthält.

Aussagenlogische Resolution

Lemma 2.51.

Sei $\mathcal{C}$ eine Klauselmenge. Seien $C_1, C_2 \in \mathcal{C}$ und $C \in \mathit{Res}(C_1, C_2)$.

Dann gilt $\{C_1, C_2\} \models C$ und $\mathcal{C}$ und $\mathcal{C} \cup \{C\}$ sind äquivalent.

Beweis (Teil 1). Wir zeigen zunächst, dass $\{C_1, C_2\} \models C$.

Wir müssen also zeigen, dass jede Belegung β mit $\beta \models \{C_1, C_2\}$ auch C erfüllt.

Sei also β so, dass $\beta \models \{C_1, C_2\}$.

Sei L das resolvierte Literal, d.h. $C := \left(C_1 \setminus \{L\}\right) \cup \left(C_2 \setminus \{\bar{L}\}\right)$.

1. Angenommen, $[\![L]\!]^\beta = 1$. Da $\beta \models \{C_2\}$ folgt, dass es ein $L' \in C_2 \setminus \{\bar{L}\}$ gibt mit $[\![L']\!]^\beta = 1$. Also $\beta \models \{C\}$, da nach Definition der Resolvente $L' \in C$.
2. Sei nun $[\![L]\!]^\beta = 0$. Da $\beta \models \{C_1\}$, gibt es ein Literal $L' \in C_1 \setminus \{L\}$ mit $[\![L']\!]^\beta = 1$. Also $\beta \models \{C\}$, da nach Definition der Resolvente $L' \in C$.

Insgesamt gilt also $\beta \models C$.

Aussagenlogische Resolution

Lemma 2.52.

Sei $\mathcal{C}$ eine Klauselmenge. Seien $C_1, C_2 \in \mathcal{C}$ und $C \in \mathit{Res}(C_1, C_2)$.

Dann gilt $\{C_1, C_2\} \models C$ und $\mathcal{C}$ und $\mathcal{C} \cup \{C\}$ sind äquivalent.

Beweis (Teil 2).

Wir zeigen als nächstes, dass $\mathcal{C}$ und $\mathcal{C} \cup \{C\}$ äquivalent sind.

Dazu müssen wir zeigen, dass für alle Belegungen β gilt:

$$\beta \models \mathcal{C} \text{ gdw. } \beta \models \mathcal{C} \cup \{C\}.$$

- Wir haben bereits gesehen, dass $\{C_1, C_2\} \models C$. Da $C_1, C_2 \in \mathcal{C}$, folgt $\mathcal{C} \models \mathcal{C} \cup \{C\}$.
- Umgekehrt, wenn $\beta \models \mathcal{C} \cup \{C\}$ dann $\beta \models \mathcal{C}$.

Also sind $\mathcal{C}$ und $\mathcal{C} \cup \{C\}$ äquivalent. □

Resolutionsableitungen

Definition 2.53.

1. Eine Resolutionsableitung einer Klausel C aus eine Klauselmenge $\mathcal{C}$ ist eine Sequenz $(C_1, \ldots, C_n)$, so dass $C_n = C$ und für alle $1 \leqslant i \leqslant n$
   - $C_i \in \mathcal{C}$ oder
   - es gibt $j, k < i$ mit $C_i \in \mathit{Res}(C_j, C_k)$.

   Wir sagen, dass C einen Resolutionsbeweis aus $\mathcal{C}$ hat und schreiben dies als $\mathcal{C} \vdash_R C$.
2. Eine Resolutionswiderlegung einer Klauselmenge $\mathcal{C}$ ist eine Resolutionsableitung der leeren Klausel $\square$.

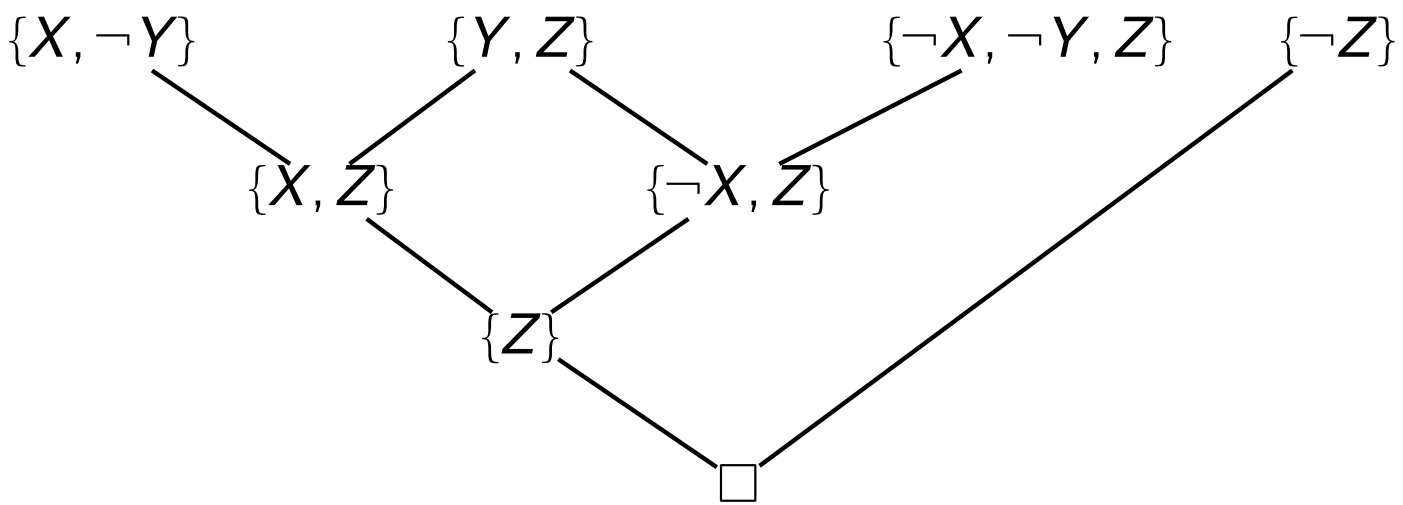
Beispiel. Sei $\mathcal{C} := \big\{\{X, \neg Y\}, \quad \{Y, Z\}, \quad \{\neg X, \neg Y, Z\}, \quad \{\neg Z\}\big\}$.
Dann ist

$$\Big(\begin{array}{ccccc} \{X, \neg Y\}, & \{Y, Z\}, & \{X, Z\}, & \{\neg X, \neg Y, Z\}, & \{\neg X, Z\}, \\ & & \{Z\}, & \{\neg Z\}, & \square \end{array} \Big)$$

eine Resolutionswiderlegung von $\mathcal{C}$.

Beispiel

Es ist oft hilfreich, Resolutionsableitungen graphisch darzustellen.



Der aussagenlogische Resolutionskalkül

Wir werden als nächstes zeigen, dass die Resolutionsmethode eine korrekte und auch vollständige Methode ist, um Unerfüllbarkeit von Klauselmengen, und damit auch aussagenlogischen Formeln, nachzuweisen.

Das heißt, wir werden folgenden Satz beweisen.

Theorem 2.54. Eine Menge $\mathcal{C}$ von Klauseln hat genau dann eine Resolutionswiderlegung, wenn $\mathcal{C}$ unerfüllbar ist.

Korrektheit der Resolution

Dazu zeigen wir zunächst das folgende Lemma.

Lemma 2.55. Sei $\mathcal{C}$ eine Klauselmenge und C eine Klausel.

Wenn $\mathcal{C} \vdash_R C$ dann $\mathcal{C} \models C$.

Beweis Sei $(C_1, \ldots, C_n)$ eine Resolutionsableitung von C aus $\mathcal{C}$. Per Induktion über i zeigen wir, dass $\mathcal{C} \models C_i$ für alle $1 \leqslant i \leqslant n$.

Für $i = n$ folgt somit $\mathcal{C} \models C_n = C$.

Induktionsbasis: $i = 1$. Es gilt $C_1 \in \mathcal{C}$ und somit $\mathcal{C} \models C_1$.

Induktionsschritt. Angenommen, die Behauptung gilt für $1, \ldots, i$.

Falls $C_{i+1} \in \mathcal{C}$, so gilt $\mathcal{C} \models C_{i+1}$.

Anderenfalls gibt es $j, k < i + 1$ mit $C_{i+1} \in \mathit{Res}(C_j, C_k)$.

Aus der Induktionsannahme folgt $\mathcal{C} \models C_j$ und $\mathcal{C} \models C_k$.

Nach dem vorherigen Lemma gilt $C_j, C_k \models C_{i+1}$ und somit $\mathcal{C} \models C$. $\square$

Vollständigkeit und Korrektheit der Resolution

Korollar 2.56. (Korrektheit des Resolutionskalküls)

Wenn eine Menge $\mathcal{C}$ von Klauseln eine Resolutionswiderlegung hat, ist $\mathcal{C}$ unerfüllbar.

Wir zeigen als nächstes die Umkehrung des Korollars.

Lemma 2.57. (Vollständigkeit des Resolutionskalküls)

Jede unerfüllbare Klauselmenge $\mathcal{C}$ hat eine Resolutionswiderlegung.

Dazu beweisen wir zunächst die folgende Behauptung..

Behauptung. *Sei $n \in \mathbb{N}$ und sei $\mathcal{C}$ eine unerfüllbare Klauselmenge in den Variablen $\{V_1, \ldots, V_{n-1}\}$. Dann hat $\mathcal{C}$ eine Resolutionswiderlegung.*

Beweis der Behauptung

Der Beweis der Behauptung wird per Induktion über n geführt.

Induktionsbasis $n = 1$. In diesem Fall ist $\mathcal{C}$ unerfüllbar und enthält keine Variablen. Also $\mathcal{C} := \{\square\}$ und somit existiert eine Resolutionswiderlegung.

Induktionsschritt $n \to n + 1$.

Sei $\mathcal{C}$ eine unerfüllbare Klauselmenge in den Variablen $\{V_1, \ldots, V_n\}$.

Wir definieren
$$\mathcal{C}^+ := \{C \setminus \{\neg V_n\} : C \in \mathcal{C} \text{ und } V_n \notin C\} \text{ und}$$
$$\mathcal{C}^- := \{C \setminus \{V_n\} : C \in \mathcal{C} \text{ und } \neg V_n \notin C\}.$$

D.h., man erhält $\mathcal{C}^+$ indem alle Klauseln, die V_n enthalten entfernt werden und $\neg V_n$ aus den anderen entfernt wird. $\mathcal{C}^-$ ist analog definiert.

$\mathcal{C}^+$ und $\mathcal{C}^-$ sind beide unerfüllbar. Denn wäre z.B. $\mathcal{C}^+$ erfüllbar, z.B. durch $\beta \models \mathcal{C}^+$, dann würde $\beta' := \beta \cup \{V_n \mapsto 1\}$ die Menge $\mathcal{C}$ erfüllen.

Beweis der Behauptung (Forts.)

Nach Induktionsvoraussetzung gibt es Resolutionsableitungen $(C_1, \ldots, C_s)$ und $(D_1, \ldots, D_t)$ der leeren Klausel $C_s = D_t = \square$ aus $\mathcal{C}^+$ bzw. $\mathcal{C}^-$.

Falls $(C_1, \ldots, C_s)$ schon eine Ableitung von $\square$ aus $\mathcal{C}$ ist, sind wir fertig. Anderenfalls werden Klauseln C_i benutzt, die aus $\mathcal{C}$ durch Entfernen von $\neg V_n$ entstanden sind, d.h. $C_i \cup \{\neg V_n\} \in \mathcal{C}$.

Fügen wir zu diesen Klauseln und allen Resolventen wieder $\neg V_n$ hinzu, so erhalten wir eine Ableitung $(C'_1, \ldots, C'_s)$ von $\neg V_n$ aus $\mathcal{C}$.

Analog ist entweder $(D_1, \ldots, D_t)$ bereits eine Ableitung von $\square$ aus $\mathcal{C}$ oder wir erhalten eine Ableitung $(D'_1, \ldots, D'_t)$ von $\{V_n\}$ aus $\mathcal{C}$.

Beweis der Behauptung (Forts.)

Ein weiterer Resolutionsschritt auf $\{V_n\}$ und $\{V_n\}$ ergibt dann $\square$.

In diesem Fall ist also $(C'_1, \ldots, C'_s, D'_1, \ldots, D'_t, \square)$ eine Resolutionswiderlegung von $\mathcal{C}$.

Dies schließt den Beweis der Behauptung ab. $\dashv$

Vollständigkeit des Resolutionskalküls

Behauptung. *Sei $n \in \mathbb{N}$ und sei $\mathcal{C}$ eine unerfüllbare Klauselmenge in den Variablen $\{V_1, \ldots, V_{n-1}\}$. Dann hat $\mathcal{C}$ eine Resolutionswiderlegung.*

Lemma 2.58. (Vollständigkeit des Resolutionskalküls)
Jede unerfüllbare Klauselmenge $\mathcal{C}$ hat eine Resolutionswiderlegung.

Beweis. Sei $\mathcal{C}$ eine unerfüllbare Klauselmenge.

1. Ist $\mathcal{C}$ endlich, dann enthält sie nur endlich viele Variablen und der Beweis folgt sofort aus der Behauptung.

2. Ist $\mathcal{C}$ unendlich, dann folgt aus dem Kompaktheitssatz, dass bereits eine endliche Teilmenge $\mathcal{C}' \subseteq \mathcal{C}$ unerfüllbar ist.
   Also hat $\mathcal{C}'$ eine Resolutionswiderlegung.
   Diese ist aber auch eine Resolutionswiderlegung von $\mathcal{C}$. □

Vollständigkeit und Korrektheit des Resolutionskalküls

Zusammengenommen ergeben die beiden vorherigen Aussagen also folgenden Satz.

Theorem 2.59. Eine Menge $\mathcal{C}$ von Klauseln hat genau dann eine Resolutionswiderlegung, wenn $\mathcal{C}$ unerfüllbar ist.

Der Resolutionskalkül ist also eine vollständige Methode, um Unerfüllbarkeit (und damit auch Erfüllbarkeit) nachzuweisen.

Trotz seiner abstrakten Formulierung hat das Erfüllbarkeitsproblem der Aussagenlogik wichtige algorithmische Anwendungen in der Informatik.

Ein effizientes Verfahren zur Lösung des Problems hätte daher weitreichende Auswirkungen.

Wir werden aber als nächstes zeigen, dass ein solches Verfahren vermutlich nicht existieren kann, da das Problem NP-vollständig ist.

2.8. Das aussagenlogische Erfüllbarkeitsproblem

NP-Vollständigkeit von SAT

Definition 2.60. Das aussagenlogische Erfüllbarkeitsproblem (SAT) ist das Problem

SAT

Eingabe. Eine aussagenlogische Formel $\varphi \in \mathrm{AL}$

Problem. Entscheide, ob φ erfüllbar ist.

Theorem 2.61. (Cook 1970, Levin 1973)

SAT ist NP-vollständig.

Erinnerung: NP-Vollständigkeit

Erinnerung: NP-Vollständigkeit

Wir erinnern hier kurz an den Begriff der NP-Vollständigkeit eines Berechnungsproblems P bzw. einer Sprache $L \subseteq \Sigma^*$ über einem Alphabet Σ.

Wir führen hier die Begriffe nur soweit ein, wie sie zum Beweis des Satzes von Cook und Levin nötig sind.

Für eine ausführliche Behandlung des Stoffes verweisen wir auf die Vorlesung TheGI 2.

Nicht-deterministische Turing-Maschinen

Definition 2.62.

Eine nicht-deterministische (1-Band) Turing-Maschine (NTM) ist ein Tuple $M := (Q, \Sigma, \Gamma, \Delta, q_0, F)$, wobei:

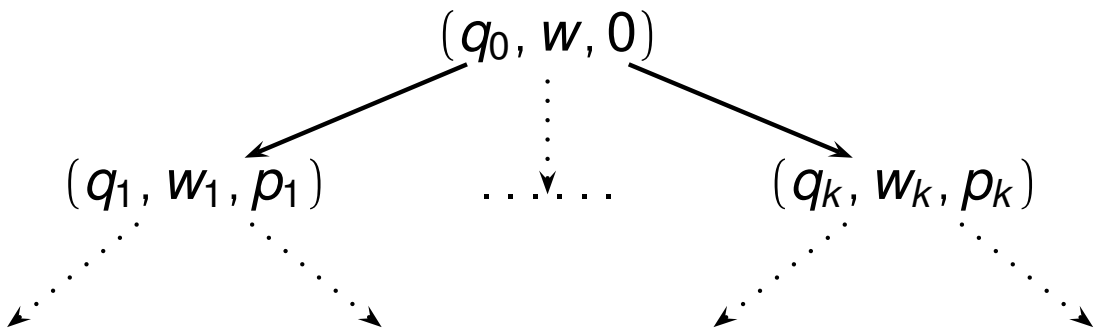
- Q ist eine endliche Menge von Zuständen
- Σ ist das endliche Eingabealphabet
- $\Gamma \supseteq \Sigma \dot\cup \{\square\}$ ist das endliche Arbeitsalphabet
- $\Delta \subseteq (Q \setminus F) \times \Gamma \times Q \times \Gamma \times \{-1, 0, 1\}$ ist die Übergangsrelation
- $q_0 \in Q$ ist der Startzustand
- $F \subseteq Q$ ist die Menge der Endzustände

Wir nehmen hier immer an, dass $\Sigma := \{0, 1\}$ und $\Gamma := \Sigma \dot\cup \{\square\}$.

Eine Konfiguration von M ist ein Tripel (q, w, p), wobei q der aktuelle Zustand, w die aktuelle Bandinschrift und p die aktuelle Position des Lese-/Schreibkopfes ist.

Der Lauf einer NTM

Die Berechnung einer nicht-deterministischen Turing-Maschine $(Q, \Sigma, \Gamma, \Delta, q_0, F)$ auf Eingabe $w \in \Sigma^*$ ist ein Berechnungsbaum:



Startkonfiguration. $(q_0, w, 0)$.

Stopkonfiguration. (q, w, p) hat keinen Nachfolger, wenn $q \in F$.

Nachfolger. (q, w, p) mit $w := w_1 \ldots w_k$ und $p \leqslant k$ hat Nachfolger $(q', w_1 \ldots w_{p-1} w' w_{p+1} \ldots w_{k'}, p')$ für alle $(q, w_p, q', w', m) \in \Delta$ wobei

$$k' := \begin{cases} k+m & \text{wenn } p = k \text{ und } m = 1 \\ k & \text{sonst} \end{cases} \qquad p' := \begin{cases} 0 & \text{wenn } p = 0 \text{ und } m = -1 \\ p+m & \textit{sonst} \end{cases}$$

Nicht-deterministische Turing-Akzeptoren

Wir interessieren uns besonders für NTMs, die Sprachen entscheiden, d.h. die Eingabe entweder akzeptieren oder verwerfen.

Nicht-deterministischer Turing-Akzeptor: $\mathcal{M} := (Q, \Sigma, \Gamma, \Delta, q_0, F_a, F_r)$

Berechnungspfad oder Lauf von $\mathcal{M}$ auf Eingabe w:

Ein Pfad von der Anfangskonfiguration $(q_0, w, 0)$ zu einer Stopkonfiguration (q_f, w', p) mit $q_f \in F_a \cup F_r$ im Berechnungsbaum.

Akzeptierender Pfad: (q_f, w, p') ist akzeptierend, d.h. $q \in F_a$.
(wir sagen auch akzeptierender Lauf)

Verwerfender Pfad. (q_f, w, p') ist verwerfend, d.h. $q \notin F_a$.

Die durch eine NTM $\mathcal{M}$ akzeptierte Sprache:

$$\mathcal{L}(\mathcal{M}) := \{w \in \Sigma^* : \text{ es existiert ein akzeptierender Lauf von } \mathcal{M} \text{ auf } w\}$$

Nicht-deterministische Zeit- und Platzkomplexität

Definition 2.63.

Sei $\mathcal{M}$ ein nicht-deterministischer Turing-Akzeptor und seien $S, T : \mathbb{N} \to \mathbb{N}$ Funktionen.

1. $\mathcal{M}$ ist T-Zeitbeschränkt, wenn sie auf jeder Eingabe $w \in \Sigma^*$ nach $\leqslant T(|w|)$ Schritten hält.

   Genauer:

   - Auf Eingabe w ist die Länge jedes Berechnungspfades von $\mathcal{M}$ höchstens $T(|w|)$

2. $\mathcal{M}$ ist S-Platzbeschränkt wenn, auf Eingabe $w \in \Sigma^*$, jeder Berechnungspfad $\leqslant S(|w|)$ Zellen benutzt.

   (Hier nehmen wir an, dass die NTM ein separates Eingabeband hat, dessen Größe wir nicht mitzählen.)

Nicht-deterministische Komplexitätsklassen

Definition 2.64. Seien $T, S : \mathbb{N} \to \mathbb{N}$ monoton wachsende Funktionen.

1. $\text{NTIME}(T)$ ist die Klasse aller Sprachen $\mathcal{L}$ für die es eine T-zeitbeschränkte NTM gibt, die $\mathcal{L}$ entscheidet.
2. $\text{NSPACE}(S)$ ist die Klasse aller Sprachen $\mathcal{L}$, für die es eine S-platzbeschränkte NTM gibt, die $\mathcal{L}$ entscheidet.

Nicht-deterministische Komplexitätsklassen

Einige wichtige nicht-deterministische Komplexitätsklasen:

- Zeitkomplexitätsklassen:
  - NP $:= \bigcup_{d \in \mathbb{N}} \text{NTIME}(n^d)$
  - NEXPTIME $:= \bigcup_{d \in \mathbb{N}} \text{NTIME}(2^{n^d})$
- Platzkomplexitätsklassen:
  - NLOGSPACE $:= \bigcup_{d \in \mathbb{N}} \text{NSPACE}(d \log n)$
  - NPSPACE $:= \bigcup_{d \in \mathbb{N}} \text{NSPACE}(n^d)$
  - NEXPSPACE $:= \bigcup_{d \in \mathbb{N}} \text{NSPACE}(2^{n^d})$

Polynomial-Time Reductions

Zur Definition der NP-Vollständigkeit brauchen wir noch den Begriff der polynomialzeit many-one Reduktionen.

Definition 2.65. Eine Sprache $\mathcal{L}_1 \subseteq \Sigma^*$ ist polynomiell reduzierbar auf $\mathcal{L}_2 \subseteq \Sigma^*$, geschrieben $\mathcal{L}_1 \leqslant_p \mathcal{L}_2$, wenn es eine polynomialzeit berechenbare Funktion $f : \Sigma^* \to \Sigma^*$ gibt, so dass für alle $w \in \Sigma^*$

$$w \in \mathcal{L}_1 \qquad \Longleftrightarrow \qquad f(w) \in \mathcal{L}_2.$$

Lemma 2.66.

Wenn $\mathcal{L}_1 \leqslant_p \mathcal{L}_2$ und $\mathcal{L}_2 \in \text{PTIME}$, dann auch $\mathcal{L}_1 \in \text{PTIME}$.

Wir brauchen meistens die Kontraposition: Wenn $\mathcal{L}_1 \leqslant_p \mathcal{L}_2$ und $\mathcal{L}_1 \notin \text{PTIME}$, dann auch $\mathcal{L}_2 \notin \text{PTIME}$.

NP-Vollständigkeit

Definition. Sei Σ ein Alphabet. Ein Problem $L \subseteq \Sigma^*$ ist NP-vollständig, wenn es in NP ist und jedes andere Problem in NP auf L in polynomieller Zeit reduziert werden kann.

- In TheGI 2 haben Sie schon gesehen, wie NP-Vollständigkeit eines Problems $P \in$ NP dadurch gezeigt werden kann, dass man ein anderes NP-vollständiges Problem auf P reduziert.
- Die Herausforderung besteht darin, ein erstes Problem zu finden, von dem man NP-Vollständigkeit ohne Reduktionen nachweisen kann.
- Dies gelang Ende der 60er Jahre Stephen Cook für das SAT Problem, der für seine bahnbrechenden Arbeiten zur NP-Vollständigkeit den Turing-Award erhielt.

Der Satz von Cook und Levin

Der Satz von Cook und Levin

Theorem 2.67. (Cook 1970, Levin 1973)

SAT ist NP-vollständig.

Beweis.

1. $\text{SAT} \in \text{NP}$

   Eine NTM kann einfach in Polynomialzeit eine Belegung der Variablen raten und verifizieren, dass sie die Eingabeformel erfüllt.

2. Zur NP-Vollständigkeit müssen wir noch zeigen: für jede Sprache $\mathcal{L} \in \text{NP}$,

   $$\mathcal{L} \leqslant_p \text{SAT}$$

   Sei also $\mathcal{L} \in \text{NP}$ mit $\mathcal{L} \subseteq \Sigma^*$.

   Für jede Eingabe $w \in \Sigma^*$ konstruieren wir eine aussagenlogische Formel $\varphi_{\mathcal{L},w}$, so dass

   $$w \in \mathcal{L} \iff \varphi_{\mathcal{L},w} \text{ erfüllbar.}$$

Beweis des Satzes von Cook und Levin

Da $\mathcal{L} \in \mathrm{NP}$ existiert eine 1-Band NTM $\mathcal{M} := (Q, \Sigma, \Gamma, q_0, \Delta, F_a, F_r)$ die $\mathcal{L}$ in Zeit $p(n)$, für ein Polynom p, akzeptiert.

Beachte. Aus Eingaben der Länge n ist die Länge jedes Berechnungspfads sowie die Länge jeder Konfiguration eines Pfades von $\mathcal{M}$ durch $p(n)$ beschränkt.

Idee. Wir beschreiben den Lauf von $\mathcal{M}$ auf Eingabe w durch eine Formel.

Beweis des Satzes von Cook und Levin

Beschreiben einer Konfiguration: Wir repräsentieren eine Konfiguration $(q, p, a_0 \dots a_{p(n)})$ durch eine Menge

$$\overline{C} := \{Q_q, P_i, S_{a,i} : \quad q \in Q, \quad a \in \Gamma \quad 0 \leqslant i < p(n)\}$$

von Variablen und die Wahrheitsbelegung β mit

$$\beta(Q_s) := \begin{cases} 1 & s = q \\ 0 & s \neq q \end{cases} \qquad \beta(P_s) := \begin{cases} 1 & s = p \\ 0 & s \neq p \end{cases} \qquad \beta(S_{a,i}) := \begin{cases} 1 & a = a_i \\ 0 & a \neq a_i \end{cases}$$

Variablen.

Q_q: Für alle $q \in Q$

Bedeutung: $\mathcal{M}$ ist im Zustand $q \in Q$

P_i: Für alle $0 \leqslant i \leqslant p(n)$

Bedeutung: der Kopf steht an Position i

$S_{a,i}$: Für alle $a \in \Gamma$ und $0 \leqslant i \leqslant p(n)$

Bedeutung: Bandposition i enthält Symbol a

Beweis des Satzes von Cook und Levin

Man beachte die folgende Formel $\textsc{Conf}(\overline{C})$ mit Variablen

$$\overline{C} := \left\{ Q_q, P_i, S_{a,i} : \quad q \in Q, \quad a \in \Gamma, \quad 0 \leqslant i < p(n) \right\}$$

$$\textsc{Conf} := \bigvee_{q \in Q} \left(Q_q \wedge \bigwedge_{q' \neq q} \neg Q_{q'} \right) \quad \wedge \quad \bigvee_{p \leqslant p(n)} \left(P_p \wedge \bigwedge_{p' \neq p} \neg P_{p'} \right) \wedge$$

$$\bigwedge_{1 \leqslant i \leqslant p(n)} \bigvee_{a \in \Gamma} \left(S_{a,i} \wedge \bigwedge_{b \neq a \in \Gamma} \neg S_{b,i} \right)$$

Definition 2.68. Für jede Belegung β von $\overline{C}$ definieren wir $\mathrm{conf}(\overline{C}, \beta)$ als

$$\left\{ (q, p, w_1 \dots w_{p(n)}) : \beta(Q_q) = 1, \beta(P_p) = 1, \beta(S_{w_i,i}) = 1, 0 \leqslant i \leqslant p(n) \right\}$$

Lemma 2.69. Wenn $\beta \models \textsc{Conf}(\overline{C})$, dann gilt $|\mathrm{conf}(\overline{C}, \beta)| = 1$.

Beweis des Satzes von Cook und Levin

Definition. Für jede Belegung β von $\overline{C}$ definieren wir $\mathrm{conf}(\overline{C}, \beta)$ als

$$\left\{ (q, p, w_0 \dots w_{p(n)}) : \begin{array}{l} \beta(Q_q) = 1, \\ \beta(P_p) = 1, \\ \beta(S_{w_i,i}) = 1 \text{ für alle } 0 \leqslant i \leqslant p(n) \end{array} \right\}$$

Lemma. Wenn $\beta \models \textsc{Conf}(\overline{C})$ erfüllt, dann $|\mathrm{conf}(\overline{C}, \beta)| = 1$.

Bemerkung. β kann auch für weitere Variablen neben $\overline{C}$ definiert sein.

Notation. Wir schreiben $\mathrm{conf}(\overline{C}, \beta) := (q, p, w)$.

$\mathrm{conf}(\overline{C}, \beta)$ ist eine potentielle Konfiguration von $\mathcal{M}$, aber eventuell ist sie nicht von der Startkonfiguration von $\mathcal{M}$ auf Eingabe w erreichbar.

Umgekehrt: Jede Konfiguration $(q, p, w_1 \dots w_{p(n)})$ induziert eine erfüllende Belegung β von conf.

Beweis des Satzes von Cook und Levin

Betrachte die folgende Formel $\textsc{Next}(\overline{C}, \overline{C}')$ definiert als

$$\textsc{Conf}(\overline{C}) \wedge \textsc{Conf}(\overline{C}') \wedge \textsc{Nochange}(\overline{C}, \overline{C}') \wedge \textsc{Change}(\overline{C}, \overline{C}').$$

$$\textsc{Nochange} := \bigvee_{0 \leqslant p \leqslant p(n)} \left(P_p \wedge \bigwedge_{\substack{i \neq p \\ a \in \Gamma}} (S_{a,i} \to S'_{a,i}) \right)$$

$$\textsc{Change} := \bigvee_{0 \leqslant p \leqslant p(n)} \left(P_p \wedge \bigvee_{\substack{q \in Q \\ a \in \Gamma}} (Q_q \wedge S_{a,p} \wedge \bigvee_{(q,a,q',b,m) \in \Delta} (Q'_{q'} \wedge S'_{b,p} \wedge P'_{p+m})) \right)$$

Lemma 2.70. Für jede Belegung β, die auf $\overline{C}, \overline{C}'$ definiert ist:

$$\beta \text{ erfüllt } \textsc{Next}(\overline{C}, \overline{C}') \iff \text{conf}(\overline{C}, \beta) \vdash_{\mathcal{M}} \text{conf}(\overline{C}', \beta)$$

Beweis des Satzes von Cook und Levin

Bisher definiert:

- $\textsc{Conf}(\overline{C})$: $\overline{C}$ beschreibt eine potentielle Konfiguration
- $\textsc{Next}(\overline{C}, \overline{C}')$: $\text{conf}(\overline{C}, \beta) \vdash_{\mathcal{M}} \text{conf}(\overline{C}', \beta)$

Startkonfiguration. Sei $w := w_0 \dots w_{n-1} \in \Sigma^*$ eine Eingabe von $\mathcal{M}$

$$\textsc{Start}_{\mathcal{M},w}(\overline{C}) := \textsc{Conf}(\overline{C}) \wedge Q_{q_0} \wedge P_0 \wedge \bigwedge_{i=0}^{n-1} S_{w_i,i} \wedge \bigwedge_{i=n}^{p(n)} S_{\square,i}$$

Eine Belegung β erfüllt $\textsc{Start}_{\mathcal{M},w}(\overline{C})$ gdw. $\overline{C}$ die Startkonfiguration von $\mathcal{M}$ auf Eingabe w repräsentiert.

Akzeptierende Stopkonfiguration.

$$\textsc{Acc-Conf}(\overline{C}) := \textsc{Conf}(\overline{C}) \wedge \bigvee_{q \in F_a} Q_q$$

Belegung β erfüllt $\textsc{Acc-Conf}(\overline{C})$ gdw. $\overline{C}$ repräsentiert eine akzeptierende Stopkonfiguration von $\mathcal{M}$.

Die komplette Reduktion: Variablen

Da $\mathcal{L} \in \mathrm{NP}$, existiert eine 1-Band NTM $\mathcal{M} := (Q, \Sigma, \Gamma, q_0, \Delta, F_a, F_r)$, die $\mathcal{L}$ in Zeit $p(n)$ entscheidet, für ein Polynom p.

Variablen:

$Q_{q,t}$: Für alle $q \in Q$, $0 \leqslant t \leqslant p(n)$

Bedeutung: $\mathcal{M}$ ist zur Zeit t im Zustand $q \in Q$

$P_{i,t}$: Für alle $0 \leqslant i, t \leqslant p(n)$

Bedeutung: Zur Zeit t ist der Kopf an Position i

$S_{a,i,t}$: Für alle $a \in \Sigma \dot{\cup} \{\square\}$ und $0 \leqslant i, t \leqslant p(n)$

Bedeutung: Zur Zeit t enthält Bandposition i das Symbol a

Notation. $\overline{C}_t := \{Q_{q,t},\ P_{i,t},\ S_{a,i,t} : \quad q \in Q, 0 \leqslant i \leqslant p(n), \quad a \in \Gamma\}$

Die komplette Reduktion: Die Formeln

Gegeben:

1-Band NTM $\mathcal{M} := (Q, \Sigma, \Gamma, q_0, \Delta, F_a, F_r)$, die $\mathcal{L}$ in Zeit $p(n)$ akzeptiert

Eingabe $w := w_0 \dots w_{n-1}$.

Konstruiere

$$\varphi_{\mathcal{M},w} := \quad \textsc{Start}_{\mathcal{M},w}(\overline{C}_0) \wedge \qquad C_0 \text{ kodiert Startkonf.}$$

$$\bigvee_{0 \leqslant t \leqslant p(n)} \begin{cases} \textsc{Acc-Conf}(\overline{C}_t) \wedge & \mathcal{M} \text{ akzeptiert nach } t \text{ Schritten} \\ \bigwedge_{0 \leqslant i < t} \textsc{Next}(\overline{C}_i, \overline{C}_{i+1}) & \overline{C}_0, \dots, \overline{C}_t \text{ kodieren Berechn.pfad} \end{cases}$$

Bemerkung. Eine akzeptierende oder verwerfende Stopkonfiguration hat keinen Nachfolger.

Lemma 2.71. $w \in \mathcal{L} \iff \varphi_{\mathcal{M},w}$ erfüllbar.

Der Satz von Cook und Levin

Theorem 2.67. (Cook 1970, Levin 1973)

SAT ist NP-vollständig.

Beweis.

1. $\text{SAT} \in \text{NP}$

   Eine NTM kann einfach in Polynomialzeit eine Belegung der Variablen raten und verifizieren, dass sie die Eingabeformel erfüllt.

2. Zur NP-Vollständigkeit müssen wir noch zeigen: für jede Sprache $\mathcal{L} \in \text{NP}$,

   $$\mathcal{L} \leqslant_p \text{SAT}$$

   Sei also $\mathcal{L} \in \text{NP}$ mit $\mathcal{L} \subseteq \Sigma^*$.

   Für jede Eingabe $w \in \Sigma^*$ konstruieren wir eine aussagenlogische Formel $\varphi_{\mathcal{L},w}$, so dass

   $$w \in \mathcal{L} \iff \varphi_{\mathcal{L},w} \text{ erfüllbar.}$$

Der DPLL Algorithmus

Der DPLL Algorithmus

Wir werden als nächstes einen auf der Resolution basierenden Algorithmus kennen lernen, um Formeln auf Unerfüllbarkeit zu testen.

Der DPLL-Algorithmus ist benannt nach seinen Erfindern, Davis, Putnam, Logemann, Loveland.

Der Algorithmus kombiniert backtracking mit Einheitsresolution, bei der nur Resolventen gebildet werden können, wenn mindestens eine der Klauseln nur ein Literal enthält.

Varianten des DPLL-Algorithmus bilden die Basis der meisten aktuellen SAT-Löser, wie z. B. BerkMin, zChaff, etc.

Der DPLL Algorithmus

Der Algorithmus arbeitet auf Formeln $\varphi := \bigwedge_{i=1}^{n} \bigvee_{j=1}^{n_i} L_{i,j}$ in KNF.

Repr. φ als Klauselmenge: $\Phi := \big\{\{L_{1,1}, \ldots, L_{1,n_1}\}, \ldots, \{L_{n,1}, \ldots, L_{n,n_n}\}\big\}$.

ALGORITHMUS DPLL(Φ)

Eingabe. Klauselmenge $\Phi := \big\{\{L_{1,1}, \ldots, L_{1,n_1}\}, \ldots, \{L_{n,1}, \ldots, L_{n,n_n}\}\big\}$

Ausgabe. entscheide, ob Φ erfüllbar ist.

Algorithmus. Wenn Φ leer ist, gib erfüllbar zurück.
Wenn Φ die leere Klausel $\square$ enthält, gib unerfüllbar zurück.

Unit Clause/Boolean Constraint Propagation (bcp):

Solange Φ eine Einheitsklausel $\{L\}$ enthält "setze $L := 1$"
d.h. entferne alle Klauseln, die L enthalten und
entferne $\overline{L}$ aus allen anderen Klauseln

Anderenfalls, wähle Literal L, das in Φ vorkommt.

Ergibt DPLL($\Phi \cup \{\{L\}\}$) oder DPLL($\Phi \cup \{\{\overline{L}\}\}$) erfüllbar,
gib erfüllbar zurück, sonst unerfüllbar.

Beispiel

$\{V_1\, V_2\, V_7, \neg V_2\, V_1, \neg V_1\, V_3, \neg V_3 \neg V_1, \neg V_7\, V_4\, V_5, \neg V_4\, V_5, \neg V_5\, V_6, \neg V_6 \neg V_5\}$

Wähle $V_7 := 0$

$\{V_1\, V_2, \neg V_2\, V_1, \neg V_1\, V_3, \neg V_3 \neg V_1, \neg V_4\, V_5, \neg V_5\, V_6, \neg V_6 \neg V_5\}$

Wähle $V_1 := 0$

$\{V_2, \neg V_2, \neg V_4\, V_5, \neg V_5\, V_6, \neg V_6 \neg V_5\}$ bcp $V_2 := 1 \rightsquigarrow \square$

Wähle $V_1 := 1$

$\{V_3, \neg V_3, \neg V_4\, V_5, \neg V_5\, V_6, \neg V_6 \neg V_5\}$ bcp $V_3 := 1 \rightsquigarrow \square$

Wähle $V_7 := 1$

$\{\neg V_2\, V_1, \neg V_1\, V_3, \neg V_3 \neg V_1,\ V_4\, V_5, \neg V_4\, V_5, \neg V_5\, V_6, \neg V_6 \neg V_5\}$

Wähle $V_5 := 0$

$\{\neg V_2\, V_1, \neg V_1\, V_3, \neg V_3 \neg V_1,\ V_4, \neg V_4\}$ bcp $V_4 := 1 \rightsquigarrow \square$

Wähle $V_5 := 1$

$\{\neg V_2\, V_1, \neg V_1\, V_3, \neg V_3 \neg V_1,\ V_6, \neg V_6\}$ bcp $V_6 := 1 \rightsquigarrow \square$

DPLL vs. Resolution

Es besteht ein enger Zusammenhang zwischen Resolutionswiderlegungen und Widerlegungen einer Formel durch den DPLL-Algorithmus.

Dazu stellt man einen Lauf des DPLL-Algorithmus als Entscheidungsbaum dar.

Hat der Baum die Höhe h, so gibt es auch eine (baumartige) Resolutionswiderlegung gleicher Höhe.

Der DPLL-Algorithmus ist also nicht “schneller” als die Resolution.

Der DPLL-Algorithmus

In praktischen Implementierungen des Algorithmus' werden verschiedene Optimierungen verwendet.

Auswahlregel: Welches Literal wird beim Verzweigen gewählt

Conflict Analysis: Bei einem Backtracking Schritt wird der Grund der Unerfüllbarkeit (Conflict) analysiert und intelligenter zurück gesprungen.

Clause Learning: Aus der Konfliktanalyse werden neue Klauseln generiert, die zur Formel hinzugenommen werden.

Dies soll verhindern, dass in den gleichen Konflikt hineingelaufen wird.

Random restarts: Bisweilen wird ein DPLL-Lauf abgebrochen und neu angefangen. Die gelernten Klauseln bleiben erhalten.

Eine Anwendung aus der künstlichen Intelligenz

Aussagenlogik in der KI

Eine mögliche Anwendung der Aussagenlogik sind sogenannte constraint satisfaction Probleme (CSPs).

Definition. Ein Constraint Satisfaction Problem besteht aus

- einer Menge V von Variablen
- einem Wertebereich D
- und einer Menge C von constraints, d.h. Tupeln $((v_1, \ldots, v_n), R)$ wobei $R \subseteq D^n$.

Eine Lösung eines CSPs ist eine Abbildung $f : V \to D$, so dass für alle constraints $((v_1, \ldots, v_n), R) \in C$ gilt:

$$(f(v_1), \ldots, f(v_n)) \in R$$

CSPs sind eine in der künstlichen Intelligenz viel untersuchte Problemklasse, da viele natürliche Probleme als CSPs modelliert werden können.

Beispiel für ein CSP

Ein einfaches Beispiel ist die Stundenplangenerierung.

Beispiel. Eine (kleine) Schule mit nur einem Klassenzimmer hat drei Lehrer die jeweils ein Fach D, M oder P unterrichten und zwei Klassen, die in jedem der drei Fächer unterrichtet werden sollen.

Variablenmenge V. $D_1, D_2, M_1, M_2, P_1, P_2$

Bedeutung: M_1: Klasse 1 wird durch den Mathematiklehrer unterrichtet.

Wertebereich D. Mögliche Unterrichtszeiten $8, 9, 10, \ldots$.

Constraint-Menge C. Sei $R_{\neq} := \{(t, t') \in D \times D : t \neq t'\}$.

$((D_1, D_2), R_{\neq}), \quad ((M_1, M_2), R_{\neq}), \quad ((P_1, P_2), R_{\neq})$

"Kein Lehrer hält zwei Klassen gleichzeitig"

Für alle $i = 1, 2, 3$: $((M_i, D_i), R_{\neq}), \quad ((M_i, P_i), R_{\neq}), \quad ((P_i, D_i), R_{\neq})$

"Keine Klasse hat zwei Stunden gleichzeitig"

Sudoku als CSP

Ein weiteres Beispiel ist das Spiel Sudoku.

				6	8		1	
1						7		
	4	3	2			5		
3						4		
8								9
		1						6
		6			4	8	5	
		2						7
	1		5	9				

Ziel ist es, die fehlenden Positionen so mit den Zahlen 1, ..., 9 zu füllen, dass in jeder Zeile und jeder Spalte und in jedem Block jede der Zahlen 1, ..., 9 genau einmal vorkommt.

Eine Lösung durch die Aussagenlogik

Wir formalisieren ein gegebenes Sudoku in der Aussagenlogik.

Variablen. $X_{i,j}^c$ für alle $1 \leqslant i, j, c \leqslant 9$

Idee: $X_{i,j}^c$ wird mit 1 belegt, wenn in Position (i, j) die Zahl c steht.

Generische Formeln.

- $\bigwedge_{1 \leqslant i,j \leqslant 9} \bigvee_{c=1}^{9} X_{i,j}^c$ "jede Position erhält eine Zahl"
- $\bigwedge_{1 \leqslant i,j \leqslant 9} \bigwedge_{1 \leqslant c < c' \leqslant 9} \neg(X_{i,j}^c \wedge X_{i,j}^{c'})$ "jede Position erhält $\leqslant 1$ Zahl"
- Für alle $1 \leqslant i, c \leqslant 9$ und $1 \leqslant j < j' \leqslant 9$: $\neg(X_{i,j}^c \wedge X_{i,j'}^c)$
  "Keine Zeile enthält zwei gleiche Einträge"
- Für alle $1 \leqslant j, c \leqslant 9$ und $1 \leqslant i < i' \leqslant 9$: $\neg(X_{i,j}^c \wedge X_{i',j}^c)$
  "Keine Spalte enthält zwei gleiche Einträge"
- Die entsprechende Aussage für jeden Block:
  $\neg(X_{i,j}^c \wedge X_{i',j'}^c)$ für alle $1 \leqslant c \leqslant 9$ und i, j, i', j' mit
  - $(i, j) \neq (i', j')$ und
  - $i \div 3 = j \div 3 = i' \div 3 = j' \div 3$

Eine Lösung durch die Aussagenlogik

Die Formeln auf der vorherigen Folie beschreiben ganz allgemein Lösungen für Sudokus.

D.h., eine erfüllende Belegung entspricht einer gültigen Beschriftung eines Sudokus.

Die Menge aller erfüllenden Belegungen der Formelmenge entspricht also der Menge aller korrekt ausgefüllten Sudokus.

Möchte man nun die Lösung für ein konkret gegebenes, teilausgefülltes Sudoku finden, muss man noch explizit die vorbesetzten Felder mit in die Formelmenge aufnehmen.

Steht also, wie in dem Beispiel vorher, auf Position $(2, 1)$ eine 1, so nimmt man noch die Formel $X^1_{2,1}$ zur Formelmenge hinzu.

Dies erzwingt, dass jede erfüllende Belegung die Position $2, 1$ korrekt beschriftet.

3. Strukturen

Grenzen der Aussagenlogik

Die Aussagenlogik formalisiert das Schließen über Aussagen die entweder wahr oder falsch sein können.

Die eigentliche Bedeutung der Aussagen ist dabei irrelevant.

Um über Aussagen der folgenden Form zu sprechen, ist die Aussagenlogik also nicht geeignet:

> *Für jede relle Zahl x gibt es eine natürliche Zahl $n > x$ die größer als x ist.*

- Wir müssen über verschiedene Arten von Objekten sprechen.
- Einige Aussagen müssen für alle Objekte gelten, andere nur für einige.

Ein weiteres Beispiel dieser Art ist das Sokratesbeispiel vom Anfang der Vorlesung.

Prädikatenlogik

Wir führen eine Logik ein, in der solche Aussagen gemacht werden können.

Intuitiv haben wir

- Variablen für Elemente einer Menge von Objekten, z. B. den rellen Zahlen, anstatt nur wahr oder falsch.
- Möglichkeiten, Variablen zu vergleichen, z.B. $x < y$, $x = y$ abhängig vom Kontext.
- Verknüpfungen wie $\neg, \vee, \wedge, \rightarrow, \leftrightarrow$ um komplexere Formeln bilden zu können.
- Möglichkeiten um zu sagen, dass es ein Element gibt mit bestimmten Eigenschaften oder das alle Elemente bestimmte Eigenschaften haben.

$$\forall x \big(\mathbb{R}(x) \rightarrow \exists y (\mathbb{N}(y) \wedge x < y) \big)$$

Um dies zu erreichen, müssen wir folgendes festlegen:

- Den Kontext in dem wir arbeiten, d.h. Relationen $<, +, \ldots$ die wir verwenden dürfen $\rightsquigarrow$Strukturen
- Die logische Sprache in der wir die Eigenschaften ausdrücken wollen $\rightsquigarrow$Prädikatenlogik

3.1. Relationen

Relationen

Definition 3.1. Sei $k \geqslant 1$ und A eine Menge.

1. A^k ist die Menge aller k-Tupel von Elementen aus A.
2. Eine k-stellige Relation auf A ist eine Teilmenge von A^k.

Bemerkung. Wir erlauben auch $k = 0$.

Eine nullstellige Relation $R \subseteq A^0$ ist entweder $\emptyset$ oder $\{()\}$.

Notation. Für einige spezielle Relationen wie z.B. $<, =$ benutzen wir Infix Notation und schreiben $a = b$ oder $a < b$ anstatt $(a, b) \in =$ oder $(a, b) \in <$.

Eigenschaften binärer Relationen

Definition 3.2. Eine binäre Relation $R \subseteq A^2$ einer Menge A ist

- reflexiv, wenn $(a, a) \in R$, für alle $a \in A$.
- symmetrisch, wenn aus $(a, b) \in R$ immer $(b, a) \in R$ folgt, für alle $a, b \in A$.
- antisymmetrisch, wenn $(a, b) \in R$ und $(b, a) \in R$ zusammen $a = b$ impliziert, für alle $a, b \in A$.
- transitiv, wenn aus $(a, b) \in R$ und $(b, c) \in R$ immer $(a, c) \in R$ folgt, für alle $a, b, c \in A$.

Äquivalenzrelationen

Definition 3.3. Eine Äquivalenzrelation ist eine binäre Relation, die reflexiv, transitiv und symmetrisch ist.

Beispiel. Einige Beispiel für Äquivalenzrelationen

- Gleichheit. Für jede Menge A

  $$\{(a, a) \in A^2 : a \in A\}$$

- Gleichmächtigkeit. Für jede Menge A

  $$\{(A, B) \in \mathcal{P}(A)^2 : A, B \text{ haben die gleiche Kardinalität } \}$$

- Logische Äquivalenz.

  $$\{(\varphi, \psi) \in \mathsf{AL}^2 : \varphi \equiv \psi\}$$

Ordnungen

Definition 3.4. Sei A eine Menge.

1. Eine (strikte) partielle Ordnung $<$ über eine Menge A ist eine irreflexive und transitive binäre Relation über A.
2. Eine (strikte) lineare Ordung $<$ über A ist eine partielle Ordung über A, so dass für alle $a, b \in A$:

$$a < b, \quad a = b \quad \text{or } b < a \qquad (*)$$

Bemerkungen.

- Bedingung 1 impliziert, dass $<$ antisymmetrisch ist.
- Bisweilen ist es nützlich, reflexive Ordungen zu betrachten. Wir definieren $\leqslant$ als $< \cup \{(a, a) \in A^2 : a \in A\}$.

  Das heißt, eine reflexive lineare Ordung ist eine reflexive, antisymmetrische, transitive binäre Relation für die $(*)$ gilt.

Graphen und gerichtete Graphen

Graphen und gerichtete Graphen

Definition 3.5. Ein gerichteter Graph G ist ein Paar $G := (V, E)$, wobei

- V eine Menge ist und
- $E \subseteq V^2$ eine binäre irreflexive Relation über V ist.

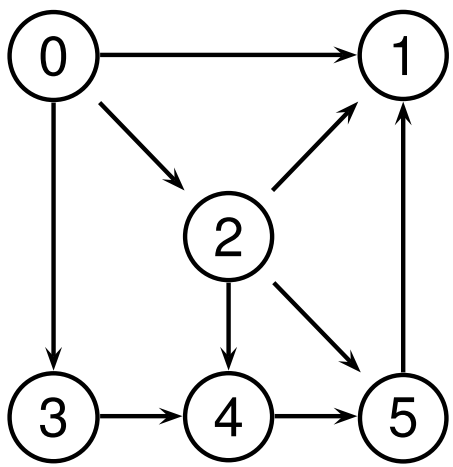
Die Elemente von V werden Knoten und die Elemente von E Kanten genannt.

Beispiel. Sei $G := (V, E)$ mit

$$V := \{0, 1, 2, 3, 4, 5\}$$

$$E := \{(0,1), (0,2), (0,3), (2,1), (2,4), (2,5), (3,4), (4,5), (5,1)\}.$$

Wir werden Graphen wie üblich graphisch darstellen.

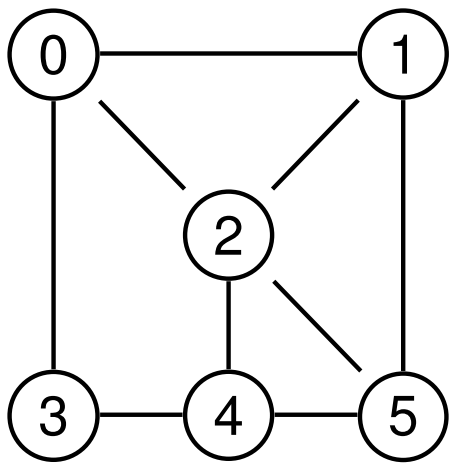


Ungerichtete Graphen

Definition 3.6. Ein ungerichteter Graph, oder einfach Graph, ist ein Paar $G := (V, E)$, so dass

- (V, E) ein gerichteter Graph ist und
- E ist symmetrisch.

Beispiel.



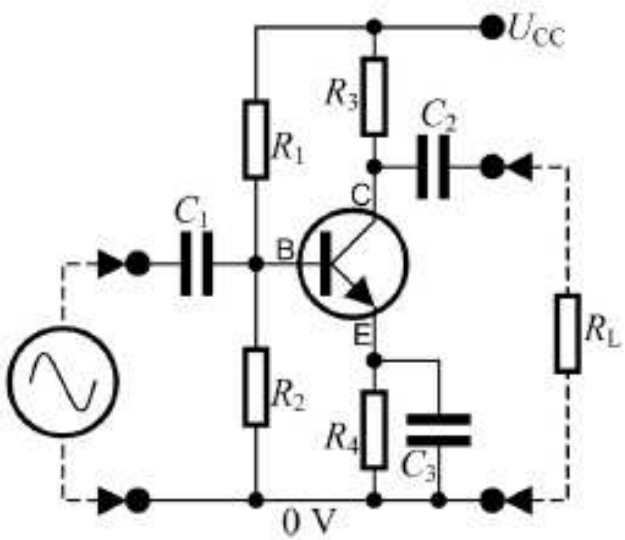
Beispiel für Graphen

Beispiel. Straßenkarten, Flugverbindungen ...

Beispiele von Graphen

Beispiele. Elektrische Schaltkreise

Knoten repräsentieren Komponenten wie z.B. Dioden, Transistoren, Widerstände .... und Kanten repräsentieren die Drähte.



Beispiele. Digitale Schaltkreises

Knoten repräsentieren Gatter und Kanten deren Verbindungen.

Beispiele für Graphen

Computer Netzwerke.

Knoten repräsentieren Computer und Kanten die Netzwerkverbindungen.

Das World Wide Web.

Knoten repräsentieren Webseiten und Kanten deren Hyperlinks.

Wege, Pfade und Kreise

Definition 3.7. Sei $G := (V, E)$ ein gerichteter Graph.

1. Ein Weg in G ist ein Tupel $(v_0, \ldots, v_l) \in V^{l+1}$, für ein $l \in \mathbb{N}$, so dass

$$(v_{i-1}, v_i) \in E \qquad \text{für alle } 1 \leqslant i \leqslant l.$$

   $(v_0, \ldots, v_l) \in V^{l+1}$ ist ein Weg von v_0 zu v_l. l ist die Länge des Wegs.

   Bemerkung. Das Tupel (v) ist ein Weg der Länge 0, für alle $v \in V$.

2. Ein Pfad in G ist ein Weg $(v_0, \ldots, v_l) \in V^{l+1}$ so dass $v_i \neq v_j$ für alle $0 \leqslant i < j \leqslant l$.

3. Ein geschlossener Weg in G ist ein Weg $(v_0, \ldots, v_l) \in V^{l+1}$ mit $v_0 = v_l$.

4. Ein Kreis oder Zykel in G ist ein Weg $(v_0, \ldots, v_l) \in V^{l+1}$ so dass $v_0 = v_l$ und $v_i \neq v_j$ für alle $1 \leqslant i < j \leqslant l$.

3.2. Strukturen

Signaturen

Definition 3.8. Eine Signatur ist eine Menge σ von Relationssymbolen, Funktionssymbolen und Konstantensymbolen.

Jedes Relationssymbol $R \in \sigma$ und jedes Funktionssymbol $f \in \sigma$ hat eine Stelligkeit

$$ar(R) \in \mathbb{N} \text{ bzw. } ar(f) \in \mathbb{N}.$$

Notation 3.9.

- Wir verwenden griechische Symbole σ, τ für Signaturen.
- Für Relationssymbole verwenden wir $R, P, Q, R', <, \leqslant, ...$
- Für Funktionssymbole verwenden wir $f, g, h, +, *...$
- Für Konstantensymbole verwenden wir $c, d, 0, 1, \ldots$

Strukturen

Definition 3.10. Sei σ eine Signatur.

Eine σ-Struktur $\mathcal{A}$ besteht aus

- einer nicht-leeren Menge A, dem Universum von $\mathcal{A}$
- eine k-stellige Relation $R^{\mathcal{A}} \subseteq A^k$ für jedes k-stellige Relationssymbol $R \in \sigma$
- eine k-stellige Funktion $f^{\mathcal{A}} : A^k \to A$ für jedes k-stelliges Funktionssymbol $f \in \sigma$
- ein Element $c^{\mathcal{A}} \in A$ für jedes Konstantensymbol $c \in \sigma$.

Bemerkung. Man beachte den Unterschied zwischen einem Symbol

$$R \in \sigma \qquad \text{oder} \qquad f \in \sigma$$

und seiner Interpretation

$$R^{\mathcal{A}} \qquad \text{bzw.} \qquad f^{\mathcal{A}}$$

in einer σ-Struktur $\mathcal{A}$.

Struktur

Notation. Wir verwenden kalligraphische Buchstaben $\mathcal{A}, \mathcal{B}, \ldots$ für Strukturen und entsprechende lateinische Buchstaben $A, B \ldots$ für deren Universen.

Wir schreiben σ-Strukturen oft als Tupel

$$\mathcal{A} := \left(A, (R^{\mathcal{A}})_{R \in \sigma}\right)$$

oder, falls $\sigma := \{R_1, \ldots, R_n\}$ endlich ist, auch

$$\mathcal{A} := \left(A, R_1^{\mathcal{A}}, \ldots, R_n^{\mathcal{A}}\right)$$

Bemerkung. In der Logik werden Strukturen meistens mit deutschen Buchstaben bezeichnet:

$\mathfrak{A}$ $\mathfrak{B}$ $\mathfrak{C}$

Beispiele von Strukturen

Beispiel: Arithmetische Strukturen

Sei $\sigma_{ar} := \{+, *, 0, 1\}$ die Signatur der Arithmetik, wobei

- $+, *$ binäre Funktionsymbole und
- $0, 1$ Konstantensymbole sind.

Wir können σ-Strukturen zur Modellierung von Körpern, etc. benutzen.

Wir definieren eine σ_{ar}-Struktur $\mathcal{N} := (\mathbb{N}, +^{\mathcal{N}}, *^{\mathcal{N}}, 0^{\mathcal{N}}, 1^{\mathcal{N}})$ mit Universum $\mathbb{N}$, wobei

- $+^{\mathcal{N}}$ und $*^{\mathcal{N}}$ die Addition und Multiplikation der natürlichen Zahlen sind und
- $0^{\mathcal{N}} := 0$ und $1^{\mathcal{N}} := 1$.

Eine andere σ_{ar}-Struktur ist $\mathcal{Z} := (\mathbb{Z}, +^{\mathcal{Z}}, *^{\mathcal{Z}}, 0^{\mathcal{Z}}, 1^{\mathcal{Z}})$ mit Universum $\mathbb{Z}$ und

- $+^{\mathcal{Z}}$ und $*^{\mathcal{Z}}$ als Addition und Multiplikation der ganzen Zahlen und
- $0^{\mathcal{Z}} := 0$ und $1^{\mathcal{Z}} := 1$.

Beispiel: Arithmetische Strukturen

Bemerkung.

σ_{ar}-Strukturen müssen nicht "natürliche" arithmetische Strukturen wie die rellen oder ganzen Zahlen sein.

Wir können genauso eine σ_{ar}-Struktur

$$\mathcal{A} := (\mathbb{N}, +^{\mathcal{A}}, *^{\mathcal{A}}, 0^{\mathcal{A}}, 1^{\mathcal{A}})$$

mit Universum $\mathbb{N}$ definieren, wobei

- $+^{\mathcal{A}}(a, b) := a^2 + b^2$,
- $*^{\mathcal{A}}$ ist die übliche Addition der natürlichen Zahlen und
- $0^{\mathcal{A}} := 17$ sowie $1^{\mathcal{A}} := 0$.

Graphen als Strukturen

Definition 3.11. Sei $\sigma_{Graph} := \{E\}$ die Signatur der Graphen.

Mit jedem gerichteten Graph (V, E) assoziieren wir eine σ_{Graph}-Struktur $\mathcal{G} := (G, E)$ mit

- $G := V$
- $E^{\mathcal{G}} := E$.

Notation. Für Graphen und deren Strukturen $\mathcal{G}$ weichen wir von der Konvention ab und bezeichnen das Universum von $\mathcal{G}$ als V.

Gefärbte Graphen

Wir betrachten oft auch gefärbte Graphen, wobei jeder Knoten mit einer Farbe aus einer festen Menge $\mathcal{C}$ von Farben gefärbt sein kann.

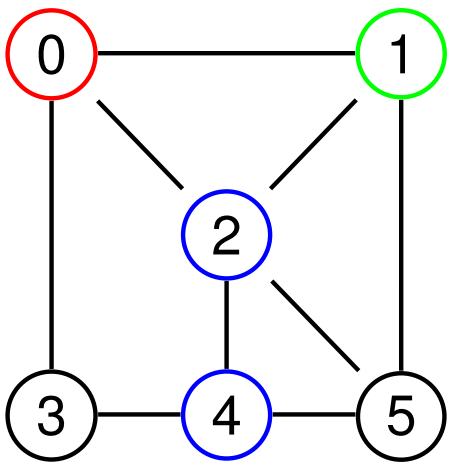
Sei $\mathcal{C}$ eine endliche Menge und sei $\sigma := \{E\} \cup \mathcal{C}$, wobei wir annehmen, dass $E \notin \mathcal{C}$.

Wir modellieren $\mathcal{C}$-gefärbte Graphen (V, E) als Strukturen

$$\mathcal{G} := (G, E^{\mathcal{G}}, (C^{\mathcal{G}})_{C \in \mathcal{C}})$$

wobei $C^{\mathcal{G}}$ alle Knoten mit Farbe C enthält.

Beispiel. Sei $\mathcal{C}$:= { Rot, Grün, Blau }.



Substrukturen und Äquivalenz zwischen Strukturen

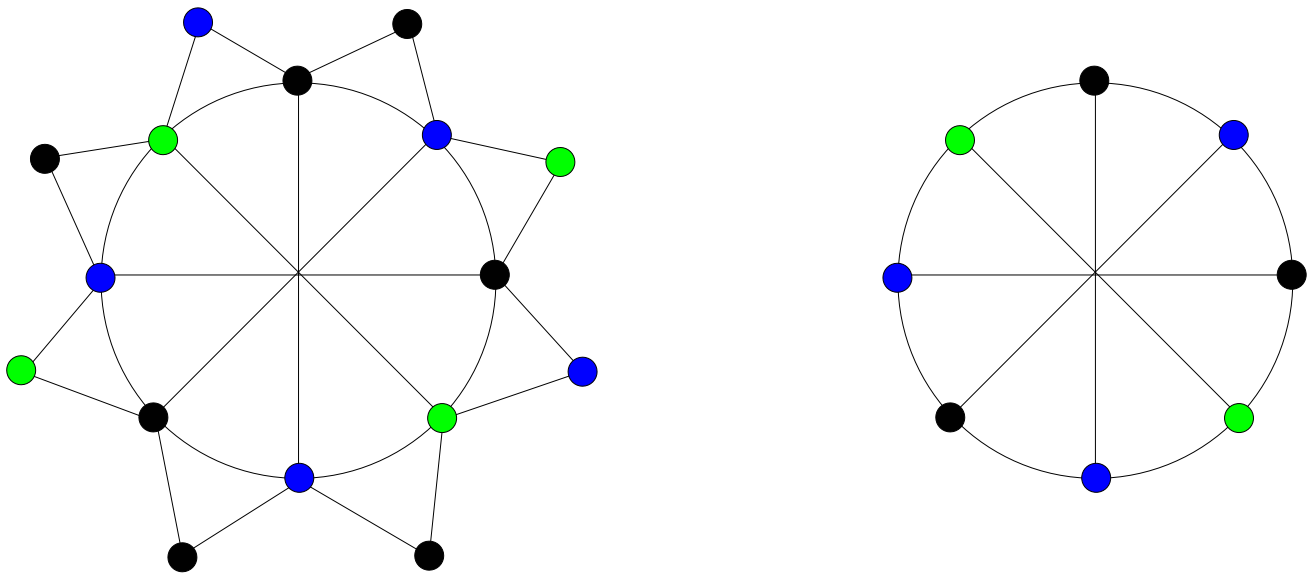
Substrukturen

Definition 3.12. Sei τ eine Signatur und seien $\mathcal{A}, \mathcal{B}$ τ-Strukturen.

1. $\mathcal{A}$ ist eine Substruktur von $\mathcal{B}$, geschrieben als $\mathcal{A} \subseteq \mathcal{B}$, wenn $A \subseteq B$ und
   - für alle k-stelligen Relationssymbole $R \in \tau$ und alle $\overline{a} \in A^k$,
     $$\overline{a} \in R^{\mathcal{A}} \quad \text{gdw.} \quad \overline{a} \in R^{\mathcal{B}}$$
   - für alle k-stelligen Funktionssymbole $f \in \tau$ und alle $\overline{a} \in A^k$, $f^{\mathcal{A}}(\overline{a}) = f^{\mathcal{B}}(\overline{a})$
   - für alle Konstantensymbole $c \in \tau$, $c^{\mathcal{A}} = c^{\mathcal{B}}$.
2. Wenn $\mathcal{A}$ eine Substruktur von $\mathcal{B}$ ist, dann ist $\mathcal{B}$ eine Erweiterung von $\mathcal{A}$.

Induzierte Substrukturen

Beispiel. A $\sigma := \{E, \text{BLUE}, \text{GREEN}\}$-structure and a sub-structure.



Hinweis. Wenn $\mathcal{A} \subseteq \mathcal{B}$, dann ist A τ-abgeschlossen, d.h. $f(\overline{a}) \in A$ für alle k-stelligen Funktionssymbole $f \in \tau$ und alle $\overline{a} \in A^k$ und $c^{\mathcal{B}} \in A$ für alle Konstantensymbole $c \in \tau$.

Umgekehrt gibt es für jede τ-abgeschossene Menge $\mathbf{A} \subseteq B$ genau eine Substruktur $\mathcal{A} \subseteq \mathcal{B}$ mit Universum A.

Dies wird die durch A induzierte Substruktur genannt.

τ-abgeschlossene Mengen

Definition 3.13. Sei τ eine Signatur und $\mathcal{B}$ eine τ-Struktur mit Universum B.

Eine Menge $A \subseteq B$ heißt τ-abgeschlossen in $\mathcal{B}$, wenn

1. $c^{\mathcal{B}} \in A$ für alle Konstantensymbole $c \in \tau$ und
2. wenn $f \in \tau$ ein k-stelliges Funktionssymbol ist und $\overline{a} \in A^k$ ein k-Tupel von Elementen, so ist $f(\overline{a}) \in A$.

Proposition 3.14.
Sei τ eine Signatur und $\mathcal{B}$ eine τ-Struktur mit Universum B.

1. Ist $\mathcal{A} \subseteq \mathcal{B}$ eine Unterstruktur von $\mathcal{B}$ mit Universum A, dann ist A *tau*-abgeschlossen in $\mathcal{B}$.
2. Umgekehrt gibt es für jede in $\mathcal{B}$ τ-abgeschossene Menge $\mathbf{A} \subseteq B$ genau eine Substruktur $\mathcal{A} \subseteq \mathcal{B}$ mit Universum A.

Dies wird die durch A induzierte Substruktur genannt.

Beispiel

Beispiel.

Sei $\mathcal{Z} := (\mathbb{Z}, <^{\mathcal{Z}}, +^{\mathcal{Z}})$, wobei $<^{\mathcal{Z}}$ die natürliche Ordnung auf $\mathbb{Z}$ und $+^{\mathcal{Z}}$ die Addition auf den ganzen Zahlen.

$\mathcal{N} := (\mathbb{N}, <^{\mathcal{N}}, +^{\mathcal{N}})$, wobei $<^{\mathcal{N}}$ die natürliche Ordnung auf $\mathbb{N}$ und $+^{\mathcal{N}}$ die übliche Addition auf den natürlichen Zahlen, ist die durch $\mathbb{N}$ induzierte Substruktur von $\mathcal{Z}$.

Expansionen und Redukte

- Substrukturen sind eine Art, in der eine Struktur in einer anderen enthalten sein kann.
- Hier haben wir ein kleineres Universum, aber die gleichen Relations-, Funktions- und Konstantensymbole.
- Eine andere Art, in der $\mathcal{A}$ in $\mathcal{B}$ enthalten sein kann, ist indem weniger Symbole zur Verfügung stehen, aber das Universum gleich bleibt.

Definition 3.15. Sei $\sigma \subseteq \tau$ eine Signatur und sei $\mathcal{B}$ eine τ-Struktur.

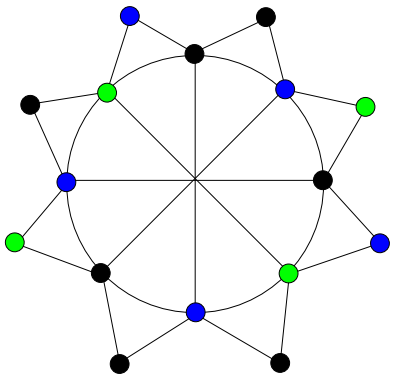
Das σ-Redukt $\mathcal{B}_{|\sigma}$ von $\mathcal{B}$ ist definiert als die σ-Struktur $\mathcal{B}_{|\sigma}$ die man aus $\mathcal{B}$ enthält, indem die Symbole aus $\tau \setminus \sigma$ "entfernt" werden, d.h. die Struktur mit

- Universum B und
- $S^{\mathcal{B}_{|\sigma}} = S^{\mathcal{B}}$ für jedes (Relations-, Funktions-, Konstanten-) Symbol $S \in \sigma$.

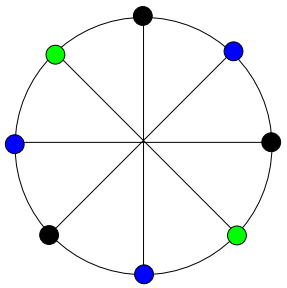
$\mathcal{B}$ heißt Expansion von $\mathcal{B}_{|\sigma}$.

Beispiel

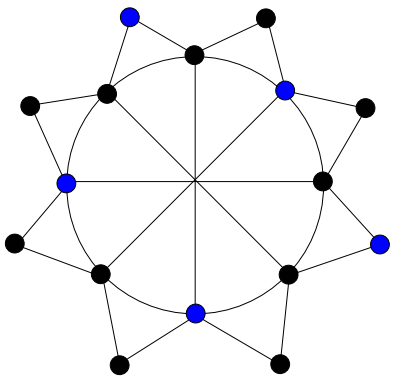
Beispiel. Eine $\sigma := \{E, \text{BLUE}, \text{GREEN}\}$-Struktur, Substruktur und Redukte.



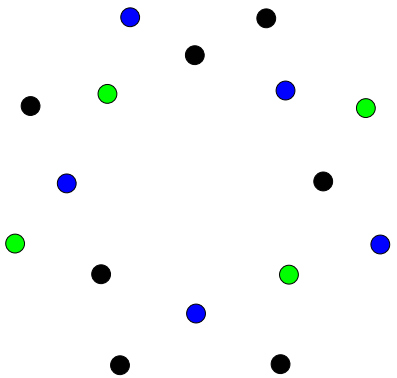
$\mathcal{G}$



$\mathcal{H} \subseteq \mathcal{G}$



$\mathcal{G}_{|\{E,\text{BLUE}\}}$

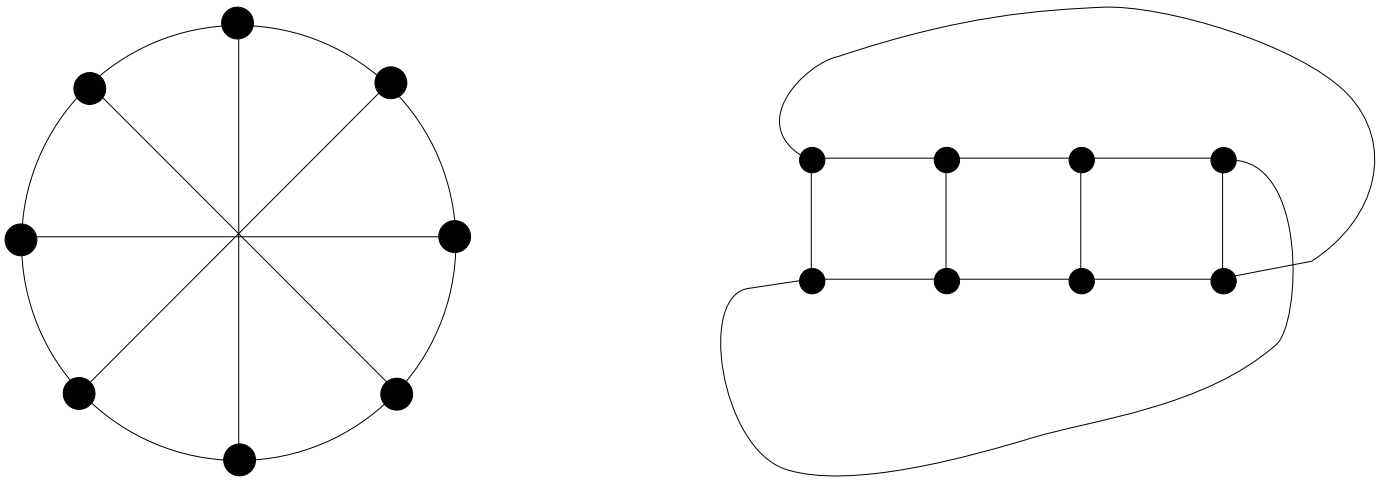


$\mathcal{G}_{|\{\text{BLUE},\text{GREEN}\}}$

3.4. Homomorphie und Isomorphie

Wann sind zwei Strukturen gleich?

Frage. Sind die folgenden zwei Graphen verschieden?



Mögliche Antworten.

Ja wenn wir daran interessiert sind, wie sie gezeichnet sind.

Nein wenn wir uns nur für ihre Knoten und Verbindungen dazwischen interessieren.

Als Strukturen $\mathcal{G}_1, \mathcal{G}_2$ über $\sigma_{Graph} := \{E\}$ sind sie identisch.

Wenn wir uns für ihre Einbettung in den $\mathbb{R}^2$ interessieren, brauchen wir eine andere Struktur, die diese Informationen enthält.

Homomorphismen

Definition 3.16. Seien $\mathcal{A}, \mathcal{B}$ zwei σ-Strukturen.

Ein Homomorphismus von $\mathcal{A}$ in $\mathcal{B}$ ist eine Funktion $h : A \to B$, so dass

- für alle k-stelligen Relationssymbole $R \in \sigma$ und alle $\overline{a} := a_1, \ldots, a_k \in A^k$:

  wenn $\overline{a} \in R^{\mathcal{A}}$ dann auch $\big(h(a_1), \ldots, h(a_k)\big) \in R^{\mathcal{B}}$.

- für alle k-stelligen Funktionssymbole $f \in \sigma$ und alle $\overline{a} := a_1, \ldots, a_k \in A^k$ gilt

  $$h(f^{\mathcal{A}}(\overline{a})) = f^{\mathcal{B}}\big(h(a_1), \ldots, h(a_k)\big).$$

- für alle Konstantensymbole $c \in \sigma$ gilt

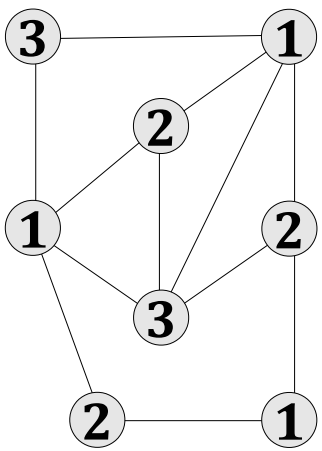
  $$h(c^{\mathcal{A}}) = c^{\mathcal{B}}.$$

Wir schreiben $h : \mathcal{A} \to_{hom} \mathcal{B}$ um zu sagen, dass h ein Homomorphismus von $\mathcal{A}$ nach $\mathcal{B}$ ist.

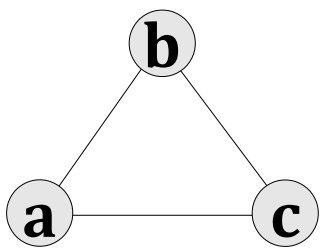
Beispiel

Wir betrachten die folgenden Graphen G und H.

G



H



Dann gilt $G \rightarrow_{hom} H$ und $H \rightarrow_{hom} G$.

Isomorphismen

Definition 3.17. Seien $\mathcal{A}, \mathcal{B}$ zwei σ-Strukturen.

Ein Isomorphismus von $\mathcal{A}$ in $\mathcal{B}$ ist eine Funktion $I : A \to B$, so dass

- I eine Bijektion zwischen A und B ist
- für alle k-stell. Relationssymb. $R \in \sigma$ und alle $\overline{a} := a_1, \ldots, a_k \in A^k$:

$$\overline{a} \in R^{\mathcal{A}} \quad \text{genau dann, wenn} \quad \big(I(a_1), \ldots, I(a_k)\big) \in R^{\mathcal{B}}.$$

- für alle k-stell. Funktionssymb. $f \in \sigma$ und alle $\overline{a} := a_1, \ldots, a_k \in A^k$ gilt

$$I(f^{\mathcal{A}}(\overline{a})) = f^{\mathcal{B}}\big(I(a_1), \ldots, I(a_k)\big).$$

- für alle Konstantensymbole $c \in \sigma$ gilt

$$I(c^{\mathcal{A}}) = c^{\mathcal{B}}.$$

Wir schreiben $I : \mathcal{A} \cong \mathcal{B}$ um zu sagen, dass I ein Isomorphismus von $\mathcal{A}$ nach $\mathcal{B}$ ist.

Iso- und Homomorphismen

Definition 3.18. Sei σ eine Signatur.

1. Zwei σ-Strukturen $\mathcal{A}, \mathcal{B}$ sind isomorph, geschrieben $\mathcal{A} \cong \mathcal{B}$, wenn es einen Isomorphismus zwischen $\mathcal{A}$ und $\mathcal{B}$ gibt.
2. Zwei σ-Strukturen $\mathcal{A}, \mathcal{B}$ sind homomorph, geschrieben $\mathcal{A} \to_{hom} \mathcal{B}$, wenn es einen Homomorphismus von $\mathcal{A}$ nach $\mathcal{B}$ gibt.

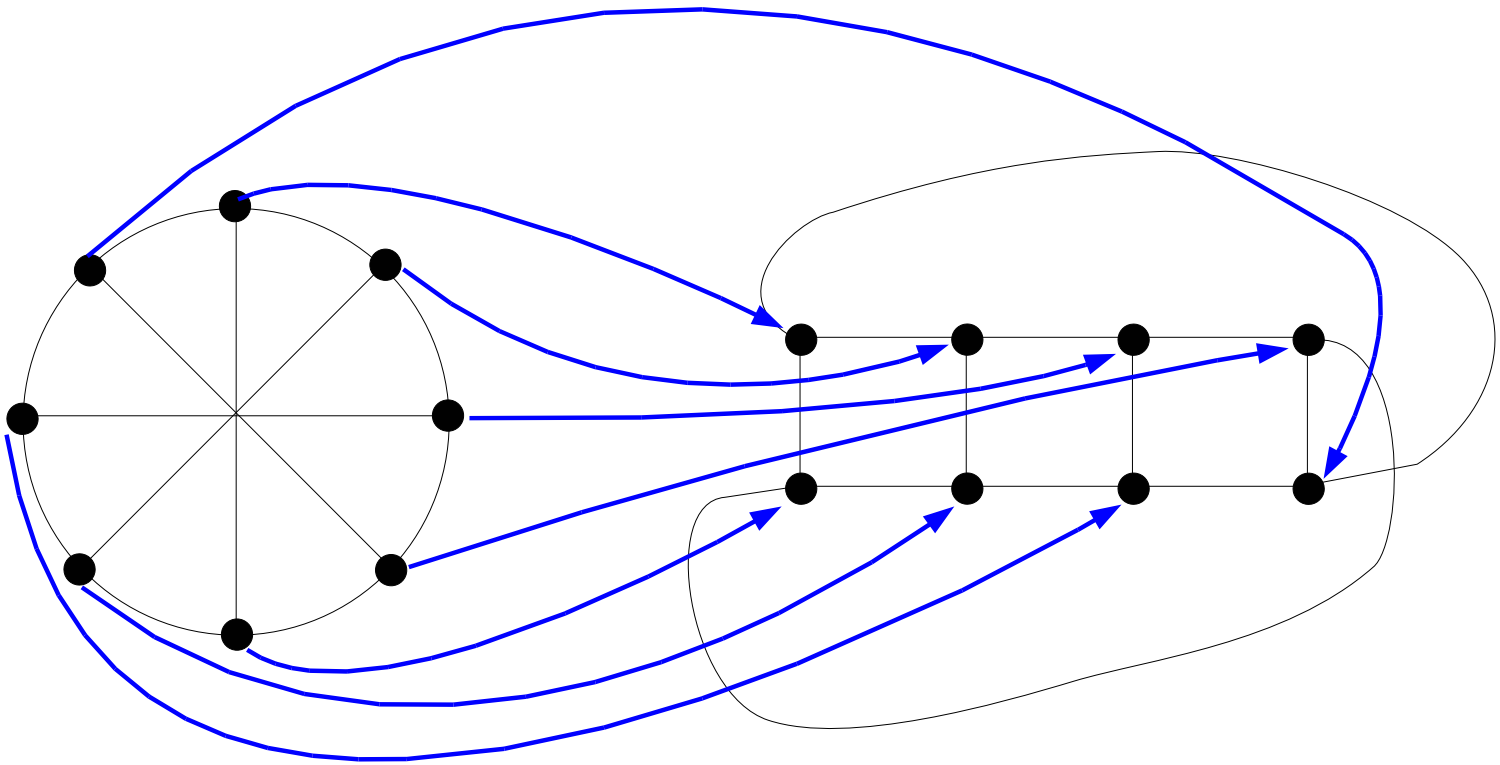
Beispiele.

- Wenn A, B endliche Mengen der gleichen Kardinalität sind, dann sind die $\emptyset$-Strukturen $(A, \emptyset) \cong (B, \emptyset)$.
- Wenn A, B endliche Mengen gleicher Kardinalität und $<^{\mathcal{A}}, <^{\mathcal{B}}$ lineare Ordnungen auf A, B sind , dann $(A, <^{\mathcal{A}}) \cong (B, <^{\mathcal{B}})$.

  Aber: $(\mathbb{Z}, <) \ncong (\mathbb{N}, <)$

Beispiel

Frage. Sind die beiden folgenden Graphen gleich?



4. Prädikatenlogik

Syntax der Prädikatenlogik

Definition 4.1. (Variablen erster Stufe)

Eine Variable erster Stufe, kurz Variable, hat die Gestalt v_i, $i \in \mathbb{N}$.

Die Menge der Variablen erster Stufe bezeichnen wir mit VAR.

Definition 4.2. Sei σ eine Signatur.

Die Menge $\mathcal{T}_\sigma$ der σ-Terme ist induktiv definiert als

Basisfall.
- $v_i \in \mathcal{T}_\sigma$ für alle $v_i \in$ VAR
- $c \in \mathcal{T}_\sigma$ für alle Konstantensymbole $c \in \sigma$

Induktionsschritt.

Ist $f \in \sigma$ ein k-stelliges Funktionssymbol und $t_1, \ldots, t_k \in \mathcal{T}_\sigma$ dann

$$f(t_1, \ldots, t_k) \in \mathcal{T}_\sigma.$$

Ein Term, in dem keine Variablen vorkommen, heißt Grundterm.

Syntax der Prädikatenlogik

Definition 4.3. Sei σ eine Signatur. Die Menge $\mathrm{FO}[\sigma]$ der prädikatenlogischen Formeln über σ ist induktiv wie folgt definiert

Basisfall.

- $t = t' \in \mathrm{FO}[\sigma]$ für alle Terme $t, t' \in \mathcal{T}_\sigma$.
- $R(t_1, \ldots, t_k) \in \mathrm{FO}[\sigma]$, für alle k-stelligen Relationssymbole $R \in \sigma$ und alle $t_1, \ldots, t_k \in \mathcal{T}_\sigma$.

Formeln der Form $t = t'$ und $R(t_1, \ldots, t_k)$ heißen atomar.

Induktionsschritt.

- Wenn $\varphi \in \mathrm{FO}[\sigma]$ dann $\neg\varphi \in \mathrm{FO}[\sigma]$.
- Wenn $\varphi, \psi \in \mathrm{FO}[\sigma]$ dann $(\varphi \vee \psi) \in \mathrm{FO}[\sigma]$, $(\varphi \wedge \psi) \in \mathrm{FO}[\sigma]$, $(\varphi \to \psi) \in \mathrm{FO}[\sigma]$ und $(\varphi \leftrightarrow \psi) \in \mathrm{FO}[\sigma]$.
- Wenn $\varphi \in \mathrm{FO}[\sigma]$ und $x \in \mathrm{VAR}$ dann $\exists x\varphi \in \mathrm{FO}[\sigma]$ und $\forall x\varphi \in \mathrm{FO}[\sigma]$.

$\mathrm{FO}[\sigma]$ heißt die Prädikatenlogik über σ oder die Sprache/Logik erster Stufe über σ.

Beispiele

Die Sprache der Graphen. Sei $\sigma_{Graph} := \{E\}$.

Die folgenden Ausdrücke sind Formeln in $\mathrm{FO}[\sigma_{Graph}]$.

- $E(x, y)$
- $\exists x \forall y E(x, y)$
- $\exists x_1 \exists x_2 \big((E(x, x_1) \wedge E(x_1, x_2)) \wedge E(x_2, y) \big)$

Die Sprache der Arithmetik und Ordnung. Sei $\sigma := \{<, +, *\}$.

Die folgenden Ausdrücke sind Formeln in $\mathrm{FO}[\sigma]$.

- $x < x + x$
- $\forall x \exists y x < y$
- $\exists x \neg \exists y \; y < x$

Bemerkung. Wir werden $<, +$ in Infixnotation verwenden, auch wenn das streng genommen keine prädikatenlogischen Formeln sind.

Freie und gebundene Variablen

Definition 4.4. Sei σ eine Signatur.

Wir schreiben $\text{var}(t)$ für die in einem σ-Term t vorkommenden Variablen.

Formal wird $\text{var}(t)$ wie folgt induktiv definiert:

- Wenn $t := v_i \in \text{VAR}$ dann $\text{var}(t) := \{v_i\}$.
- Wenn $t := c$ für ein Konstantensymbol $c \in \sigma$ dann $\text{var}(t) := \emptyset$.
- Wenn $t := f(t_1, \ldots, t_k)$ für ein k-stelliges Funktionssymbol $f \in \sigma$ und Terme $t_1, \ldots, t_k \in \mathcal{T}_\sigma$ dann $\text{var}(t) := \text{var}(t_1) \cup \cdots \cup \text{var}(t_k)$.

Freie und gebundene Variablen

Definition 4.5. Sei σ eine Signatur und sei $\varphi \in \mathrm{FO}[\sigma]$.

Die Menge $\mathit{frei}(\varphi)$ der freien Variablen von φ ist induktiv definiert durch:

- Wenn $\varphi := t_1 = t_2$, für $t_1, t_2 \in \mathcal{T}_\sigma$, dann $\mathit{frei}(\varphi) := \mathrm{var}(t_1) \cup \mathrm{var}(t_2)$.
- Wenn $\varphi := R(t_1, \ldots, t_k)$ für $R \in \sigma$ und $t_1, \ldots, t_k \in \mathcal{T}_\sigma$, dann $\mathit{frei}(\varphi) := \bigcup_{i=1}^{k} \mathrm{var}(t_i)$.
- $\mathit{frei}(\neg\varphi) := \mathit{frei}(\varphi)$ für alle $\varphi \in \mathrm{FO}[\sigma]$.
- $\mathit{frei}\big((\varphi * \psi)\big) := \mathit{frei}(\varphi) \cup \mathit{frei}(\psi)$ für alle $* \in \{\vee, \wedge, \rightarrow, \leftrightarrow\}$ und $\varphi, \psi \in \mathrm{FO}[\sigma]$.
- Wenn $\varphi := \exists x\psi$ oder $\varphi := \forall x\psi$, für $x \in \mathrm{VAR}$ and $\psi \in \mathrm{FO}[\sigma]$, dann $\mathit{frei}(\varphi) := \mathit{frei}(\psi) \setminus \{x\}$.

Eine Formel φ mit $\mathit{frei}(\varphi) := \emptyset$ heißt ein Satz.

Eine Variable, die in φ vorkommt, aber nicht frei ist, heißt gebunden.

Wir schreiben $\varphi(v_1, \ldots, v_k)$ um zu sagen, dass $\mathit{frei}(\varphi) \subseteq \{v_1, \ldots, v_k\}$.

Beispiel

Die Sprache der Graphen. Sei $\sigma_{Graph} := \{E\}$.

Die folgenden Ausdrücke sind Formeln aus $\mathrm{FO}[\sigma_{Graph}]$.

- $\varphi := E(x, y)$ $\quad$ $frei(\varphi) := \{x, y\}$
- $\varphi := \exists x \forall y (x = y \vee E(x, y))$ $\quad$ $frei(\varphi) := \emptyset$
- $\varphi := \exists x_1 \exists x_2 \big((E(x, x_1) \wedge E(x_1, x_2)) \wedge E(x_2, y) \big)$ $\quad$ $frei(\varphi) := \{x, y\}$.

Die Sprache der Arithmetik und Ordnung. Sei $\sigma := \{<, +, *\}$.

Die folgenden Ausdrücke sind Formeln aus $\mathrm{FO}[\sigma]$.

- $\varphi(x) := x < x + x$ $\quad$ $frei(\varphi) := \{x\}$
- $\varphi := \forall x \exists y x < y$ $\quad$ $frei(\varphi) := \emptyset$
- $\varphi := \exists x \neg \exists y \; y < x$ $\quad$ $frei(\varphi) := \emptyset$

4.2. Semantik der Prädikatenlogik

Belegungen

Zur Erinnerung. In der Aussagenlogik wurde die Semantik durch eine Belegung der Variablen mit Wahrheitswerten gegeben.

In der Prädikatenlogik werden wir ebenfalls Belegungen als Basis der Semantik verwenden.

Allerdings wird hier den Variablen ein Wert aus dem Universum der Struktur zugewiesen.

Definition 4.6. Sei σ eine Signatur und $\mathcal{A}$ eine σ-Struktur.

1. Eine Belegung in $\mathcal{A}$ ist eine Funktion $\beta : \mathsf{Dom}(\beta) \to A$ mit $\mathsf{Dom}(\beta) \subseteq \mathsf{VAR}$.

   β ist passend für $\varphi \in \mathsf{FO}[\sigma]$, wenn $\mathit{frei}(\varphi) \subseteq \mathsf{Dom}(\beta)$.
2. Eine σ-Interpretation ist ein Paar $(\mathcal{A}, \beta)$, bestehend aus einer σ-Struktur $\mathcal{A}$ und einer Belegung β in $\mathcal{A}$.

   $\mathcal{I} := (\mathcal{A}, \beta)$ ist passend für $\varphi \in \mathsf{FO}[\sigma]$, wenn β zu φ passt.

Definition 4.7. Sei $\mathcal{A}$ eine σ-Struktur.

1. Ist β eine Belegung, $x \in \text{VAR}$ und $a \in A$ ein Element, dann definieren wir eine neue Belegung $\beta[x/a]$ mit $\text{Dom}(\beta[x/a]) := \text{Dom}(\beta) \cup \{x\}$ durch

$$\beta[x/a](y) := \begin{cases} a & \text{wenn } x = y \\ \beta(y) & \text{sonst.} \end{cases}$$

2. Ist $\mathcal{I} := (\mathcal{A}, \beta)$ eine σ-Interpretation, $x \in \text{VAR}$ und $a \in A$ ein Element, dann definieren wir $\mathcal{I}[x/a]$ als $(\mathcal{A}, \beta[x/a])$.

Semantik der Prädikatenlogik: Terme

Definition 4.8. Sei σ eine Signatur.

Induktiv über den Formelaufbau definieren wir eine Funktion $[\![\cdot]\!]^{\cdot}$, die jedem Term $t \in \mathcal{T}_\sigma$ und jeder σ-Interpretation $\mathcal{I} := (\mathcal{A}, \beta)$ für t einen Wert $[\![t]\!]^{\mathcal{I}} \in A$ zuweist.

Basisfall.

- $[\![v_i]\!]^{\mathcal{I}} := \beta(v_i)$ für alle $v_i \in \text{VAR}$
- $[\![c]\!]^{\mathcal{I}} := c^{\mathcal{A}}$ für alle Konstantensymbole $c \in \sigma$

Induktionsschritt.

Ist $f \in \sigma$ eine k-stelliges Funktionssymbol und $t_1, \ldots, t_k \in \mathcal{T}_\sigma$ dann

$$[\![f(t_1, \ldots, t_k)]\!]^{\mathcal{I}} := f^{\mathcal{A}}\left([\![t_1]\!]^{\mathcal{I}}, \ldots, [\![t_k]\!]^{\mathcal{I}}\right).$$

Semantik der Prädikatenlogik: Formeln

Definition 4.9. Sei σ eine Signatur.

Induktiv über den Formelaufbau definieren wir eine Funktion $[\![\cdot]\!]^{\cdot}$, die jeder Formel $\varphi \in \mathrm{FO}[\sigma]$ und jeder σ-Interpretation $\mathcal{I} := (\mathcal{A}, \beta)$ für φ einen Wahrheitswert $[\![\varphi]\!]^{\mathcal{I}} \in \{0, 1\}$ zuordnet.

Basisfall.

- Für alle Terme $t, t' \in \mathcal{T}_\sigma$ definieren wir

$$[\![t = t']\!]^{\mathcal{I}} := \begin{cases} 1 & \text{wenn } [\![t]\!]^{\mathcal{I}} = [\![t']\!]^{\mathcal{I}} \\ 0 & \text{sonst.} \end{cases}$$

- Für alle k-stelligen Relationssymbole $R \in \sigma$ und alle Terme $t_1, \ldots, t_k \in \mathcal{T}_\sigma$ definieren wir

$$[\![R(t_1, \ldots, t_k)]\!]^{\mathcal{I}} := \begin{cases} 1 & \text{wenn } ([\![t_1]\!]^{\mathcal{I}}, \ldots, [\![t_k]\!]^{\mathcal{I}}) \in R^{\mathcal{A}} \\ 0 & \text{sonst.} \end{cases}$$

Semantik der Prädikatenlogik: Formeln

Induktionsschritt.

- Die Semantik der Verknüpfungen $\neg, \vee, \wedge, \rightarrow, \leftrightarrow$ ist wie in der Aussagenlogik definiert. Z.B. wenn $\varphi := \neg\psi \in \mathrm{FO}[\sigma]$ dann definieren wir

$$[\![\varphi]\!]^{\mathcal{I}} := 1 - [\![\psi]\!]^{\mathcal{I}}.$$

- Wenn $\varphi := \exists x\psi \in \mathrm{FO}[\sigma]$ dann definieren wir

$$[\![\varphi]\!]^{\mathcal{I}} := \begin{cases} 1 & \text{es gibt } a \in A \text{ so dass } [\![\psi]\!]^{\mathcal{I}[x/a]} = 1 \\ 0 & \text{sonst.} \end{cases}$$

- Wenn $\varphi := \forall x\psi \in \mathrm{FO}[\sigma]$ definieren wir

$$[\![\varphi]\!]^{\mathcal{I}} := \begin{cases} 1 & [\![\psi]\!]^{\mathcal{I}[x/a]} = 1 \text{für alle } a \in A \\ 0 & \text{sonst.} \end{cases}$$

Beispiele

Sei $\sigma := \{+, *, 0, 1\}$ die Signatur der Arithmetik und sei $\mathcal{A} := (\mathbb{N}, +^{\mathcal{A}}, *^{\mathcal{A}}, 0^{\mathcal{A}}, 1^{\mathcal{A}})$ die Struktur über den ganzen Zahlen mit der üblichen Interpretation von $+^{\mathcal{A}}, *^{\mathcal{A}}, 0^{\mathcal{A}}, 1^{\mathcal{A}}$

Sei $\beta : x \mapsto 2, y \mapsto 3$ und $\mathcal{I} := (\mathcal{A}, \beta)$.

Beispiele.

- $[\![x * x]\!]^{\mathcal{I}} := \beta(x) *^{\mathcal{A}} \beta(x) = 4$.
- $[\![x * x = y + 1]\!]^{\mathcal{I}} = 1$ falls $\beta(x) *^{\mathcal{A}} \beta(x) = \beta(y) +^{\mathcal{A}} 1^{\mathcal{A}} = 1$, was stimmt.
- Sei $\varphi(x, y) := \exists z(x * x = y + z)$. Um zu zeigen, dass $[\![\varphi]\!]^{\mathcal{I}} = 1$ müssen wir ein Element $a \in \mathbb{N}$ finden, mit $[\![x * x = y = z]\!]^{\mathcal{I} \cup \{z \mapsto a} = 1$.

  Sei $\beta' := \beta \cup \{z \mapsto 1\}$ und $\mathcal{I}' := (\mathcal{A}, \beta')$.

  Dann $[\![x * x = y + z]\!]^{\mathcal{I}'} := 1$ wie zuvor und daher $[\![\exists z(x * x = y + z)]\!]^{\mathcal{I}} = 1$

Beispiele

Die Sprache der Graphen. Sei $\sigma_{Graph} := \{E\}$.

- $\varphi := E(x, y)$ $\qquad$ $frei(\varphi) := \{x, y\}$

  Sei $\beta(x) := u$ und $\beta(y) := v$.

  $[\![\varphi]\!]^{\mathcal{I}} = 1$, wenn es eine Kante zwischen u und v gibt.

- $\varphi := \exists x \forall y \big(x = y \vee E(x, y)\big)$ $\qquad$ $frei(\varphi) := \emptyset$

  gilt in einem Graph, wenn es einen Knoten mit Kanten zu allen anderen gibt.

- $\varphi := \exists x_1 \exists x_2 \big((E(x, x_1) \wedge E(x_1, x_2)) \wedge E(x_2, y)\big)$

  $frei(\varphi) := \{x, y\}$.

  es gibt einen Pfad der Länge 3 von $\beta(x)$ zu $\beta(y)$

- $\varphi := \forall x \forall y \big(E(x, y) \rightarrow E(y, x)\big)$

  der gerichtete Graph ist eigentlich ungerichtet.

Beispiele: Ordnungen

Sei $\sigma := \{<\}$ die Signatur strikter Ordnungen.

1. "$<$ ist irreflexiv"

   für alle x gilt $x \not< x$

   $$\forall x \neg x < x$$

2. "$<$ ist transitiv"

   für alle x, y, z, wenn $x < y$ und $y < z$ dann $x < z$

   $$\forall x \forall y \forall z \big((x < y \wedge y < z) \rightarrow x < z\big)$$

3. Das heißt, wir können wie folgt sagen, dass $<$ eine partielle Ordnung ist:

   $$\varphi_{\textit{par-ord}} := \Big(\forall x \neg x < x\Big) \wedge \Big(\forall x \forall y \forall z \big((x < y \wedge y < z) \rightarrow x < z\big)\Big)$$

4. "$<$ ist total": $\varphi_t := \forall x \forall y (x < y \vee x = y \vee y < x)$

5. Eine lineare Ordung $<$ wird formalisiert durch $\varphi_{\textit{ord}} := \varphi_{\textit{par-ord}} \wedge \varphi_t$

Ein ausführliches Beispiel

Definition 4.10. Ein vertex cover eines ung. Graphs $G := (V, E)$ ist eine Menge $X \subseteq V$, s.d. für alle Kanten $e := (u, v) \in E$ mind. ein $u, v \in X$.

Problem: gegeben Graph G und $k \in \mathbb{N}$, enthält G ein vertex cover der Größe $\leqslant k$.

Kann dies in der Prädikatenlogik formalisiert werden?

Schritt 1. Schreiben Sie das Problem in deutsch auf.

G enthält ein vertex cover der Größe $\leqslant k$ wenn

- es gibt eine Menge X von $\leqslant k$ Knoten, so dass
- jede Kante (u, v) einen Endpunkt u, v in X hat.

Diese Fomalisierung benutzt

- eine Menge X über die wir in der Prädikatenlogik nicht quantifizieren können
- eine Aussage der Form für alle Kanten, was wir ebenfalls nicht benutzen können

Ein ausführliches Beispiel

Definition. Ein vertex cover eines ung. Graphs $G := (V, E)$ ist eine Menge $X \subseteq V$, s.d. für alle Kanten $e := (u, v) \in E$ mind. ein $u, v \in X$.

Problem: gegeben Graph G und $k \in \mathbb{N}$, enthält G ein vertex cover der Größe $\leqslant k$.

Wir formulieren das Problem daher um.

G enthält ein vertex cover der Größe $\leqslant k$, wenn

- es k Knoten $x_1, \ldots, x_k$ gibt, so dass
- für alle u, v: wenn es eine Kante zwischen u, v gibt, dann ist u eins der x_i oder v eins der x_i.

Das können wir nun eins-zu-eins in die Prädikatenlogik übersetzen.

$$\exists x_1 \ldots \exists x_k \; \forall u \forall v \Big(E(u, v) \rightarrow \big(\bigvee_{i=1}^{k} u = x_i \vee \bigvee_{i=1}^{k} v = x_i \big) \Big)$$

Das Koinzidenzlemma

Definition 4.11. (Koinzidenzlemma)

Seien σ, τ, τ' Signaturen, so dass $\sigma \subseteq \tau \cap \tau'$. Sei $\mathfrak{I} := (\mathcal{A}, \beta)$ eine τ-Interpretation und $\mathfrak{J} := (\mathcal{B}, \gamma)$ eine τ'-Interpretation, so dass

- $A = B$
- $S^{\mathcal{A}} = S^{\mathcal{B}}$ für alle Symbole, die in σ vorkommen.

1. Ist $t \in \mathcal{T}_\sigma$ ein σ-Term und $\beta(x) = \gamma(x)$ für alle $x \in \mathrm{var}(t)$, dann
$$[\![t]\!]^{\mathfrak{I}} = [\![t]\!]^{\mathfrak{J}}.$$
2. Ist $\varphi \in \mathrm{FO}[\sigma]$ eine Formel und $\beta(x) = \gamma(x)$ für alle $x \in \mathit{frei}(\varphi)$, dann
$$\mathfrak{I} \models \varphi \quad \Longleftrightarrow \quad \mathfrak{J} \models \varphi.$$

Notation

Notation 4.12.

1. Sei $\varphi \in \mathrm{FO}[\sigma])$ eine Formel mit $\mathit{frei}(\varphi) \subseteq \{x_1, \ldots, x_k\}$.

   Sei $\mathcal{A}$ eine σ-Struktur und β eine Belegung, so dass $\beta(x_i) := a_i$, für alle $1 \leqslant i \leqslant k$.

   Wir schreiben $\mathcal{A} \models \varphi[x_1/a_1, \ldots, x_k/a_k]$ statt $\mathcal{I} \models \varphi$.

   Dies ist möglich, da nach dem Koinzidenzlemma die Belegung der Variablen außer $x_1, \ldots, x_k$ nicht relevant ist.

2. Erinnerung: Wir schreiben $\varphi(x_1, \ldots, x_k)$ um anzudeuten, dass $\mathit{frei}(\varphi) \subseteq \{x_1, \ldots, x_k\}$.

   In diesem Fall vereinfachen wir obige Notation weiter und schreiben $\mathcal{A} \models \varphi[a_1, \ldots, a_k]$.

   Man beachte, dass dies von der Sequenz $x_1, \ldots, x_k$ abhängt.

3. Wenn φ ein Satz ist, schreiben wir $\mathcal{A} \models \varphi$.

Anwendung: Relationale Datenbanken

Beispiele: Relationale Datenbanken

Eine relationale Datenbank ist eine endliche Menge von "Tabellen".

Z. B. könnte eine Filmdatenbank wie imdb.org wie folgt aussehen

Schauspieler		
Schausp.	ID	Geburtsdatum
George Clooney	1	6. Mai 1961
Scarlett Johansson	2	22. November 1984
Jeff Daniels	3	19. Februar 1955
...	...	...

Flime		
Titel	Regie	Schau.
Good night ... und good luck	Georg Clooney	1
Good night ... und good luck	Georg Clooney	3
Lost in translation	Sofia Coppola	2
...	...	...

Die Menge τ von Tabellennamen heißt Datenbankschema.

Beispiel: Relationale Datenbanken

Relationale Datenbanken.

- Jede Spalte einer Tabelle in der Datenbank enthält Einträge vom selben Typ, z.B. Wörter oder Zahlen.

  In Datenbankterminologie werden Spaltenname Attribute genannt.

  Jedes Attribut i hat einen Typ D_i, genannt domain.
- Jede Zeile der Tabelle enthält ein Tupel $(x_1, \ldots, x_n) \in D_1 \times D_2 \times \cdots \times D_n$.

Eine Datenbanktabelle kann daher als n-stellige Relation über der Menge $D := D_1 \cup \ldots D_n$ aufgefasst werden.

Eine relationale Datenbank mit Schema τ kann also als τ-Struktur $\mathcal{D}$ wie folgt geschrieben werden:

- Das Universum $A := D$ ist die Vereinigung aller Domains.
- für jede Tabelle $R \in \sigma$ enthält die Struktur eine Relation $R^{\mathcal{D}}$, die alle Tupel der Tabelle enthält.

Beispiel: Relationale Datenbanken

Relationale Datenbanken können leicht als Struktur modelliert werden.

(Historisch wurden relationale Datenbanken nach logischen Strukturen modelliert.)

Eine Abfrage an eine Datenbank entspricht also dem Auswerten logischer Formeln in Strukturen.

Es existiert daher ein enger Zusammenhang zwischen mathematischer Logik, besonders dem Teilgebiet der endlichen Modelltheorie, und der Theorie relationaler Datenbanken.

Beispiel: Relationale Datenbanken

Beispiel. Betrachten wir noch einmal das Filmbeispiel.

Der domain aller Einträge sind Zeichenketten. Sei Σ^* die Menge aller Zeichenketten über dem Alphabet $\{a, \ldots, z, A, \ldots, Z, 0, \ldots, 9\}$.

Die Filmdatenbank entspricht folgender Struktur $\mathcal{D}$ über der Signatur $\sigma := \{ \text{Actors}, \text{Movies} \}$:

- Das Universum ist $D := \Sigma^*$
- Die Relation

$$\text{Actors}^{\mathcal{D}} := \{ \begin{array}{c} (\text{George Clooney}, 1, 6\ \text{May}\ 1961), \\ (\text{Scarlett Johansson}, 2, 22\ \text{November}\ 1984), \\ (\text{Jeff Daniels}, 3, 19\ \text{February}\ 1955) \end{array} \}$$

- Die Relation $\text{Movies}^{\mathcal{D}} := \{$ (Good night ... und good luck, Georg Clooney, 1), (Good night ... und good luck, Georg Clooney, 3), (Lost in translation, Sofia Coppola, 2) $\}$

Beispiel: Relationale Datenbanken

- Die Modellierung relationaler Datenbanken wie auf der vorherigen Folie hat die unangenehme Eigenschaft, dass das Universum unendlich ist.
- D.h., das Komplement einer Relation ist unendlich.

Datenbanken werden daher meistens als endliche Strukturen modelliert.

Das Universum enthält den active domain der Datenbank:

die Menge aller der Elemente, die in der Datenbank vorkommen.

Im Beispiel: { George Clooney, Scarlett Johansson, Jeff Daniels, 1, 2, 3, 6 May 1961, 22 November 1984, 19 February 1955, Good night ... und good luck, Lost in translation, Sofia Coppola }

Die Relationen sind wie zuvor definiert.

Beispiel: Relationale Datenbanken

Im Beispiel: D := { George Clooney, Scarlett Johansson, Jeff Daniels, Nicole Kidman, 1, 2, 3, 4, 6 May 1961, 22 November 1984, 19 February 1955, 20 June 1967, Good night ... und good luck, Lost in translation, Dogville, Sofia Coppola, Lars von Trier }

Die Datenbank entspricht also der Struktur $\mathcal{D}$ über der Signatur $\sigma := \{$ Actors, Movies $\}$ mit Universum D und

- die Relation

$$\text{Actors}^{\mathcal{D}} := \{ \begin{array}{c} \text{(George Clooney, 1, 6 May 1961),} \\ \text{(Scarlett Johansson, 2, 22 November 1984),} \\ \text{(Jeff Daniels, 3, 19 February 1955)} \\ \text{(Nicole Kidman, 4, 20 June 1967)} \end{array} \}$$

- die Relation $\text{Movies}^{\mathcal{D}} := \{$
  (Good night ... und good luck, Georg Clooney, 1),
  (Good night ... und good luck, Georg Clooney, 3),
  (Lost in translation, Sofia Coppola, 2)$\}$

Datenbankanfragen

Beispiel. In der Signatur $\sigma := \{ \text{Actors}, \text{Movies} \}$ und vorherige Datenbank:

1. $\exists x\, \exists d\, \exists n_1\, \exists n_2\, \text{Movies}(x, d, n_1) \wedge \text{Movies}(x, d, n_2) \wedge n_1 \neq n_2$

   "Es gibt einen Film mit mehr als einem Schauspieler"

2. "Es gibt einen Regisseur, der im eigenen Film mitspielt."

   $\exists d\, \exists f \exists n\, \exists b \text{Movies}(f, d, n) \wedge \text{Actors}(d, n, b)$

Die Anfragen beschreiben Eigenschaften der Datenbank, die wahr oder falsch sein können. Boolesche Anfragen

Meistens wollen wir nicht-Boolesche Informationen, z.B.

"Gib alle Filme von Georg Clooney aus"

$\varphi(f) := \exists n\, \text{Movies}(f, \text{“G Clooney”}, n)$

Die Relation $\varphi(\mathcal{A})$

Definition 4.13. Sei $\mathcal{A}$ eine σ-Struktur und $\varphi(x_1, \ldots, x_k) \in \mathsf{FO}[\sigma]$. Wir definieren

$$\varphi(\mathcal{A}) := \{(a_1, \ldots, a_k) \in A^k : \mathcal{A} \models \varphi[a_1, \ldots, a_k]\}.$$

Hinweis. Die Relation $\varphi(\mathcal{A})$ hängt nicht nur von $\mathcal{A}$ sondern auch von der Sequenz $(x_1, \ldots, x_k) \in \mathsf{VAR}^k$ ab.

Wir müssen daher diese Sequenz jeweils angeben, bevor wir die Notation benutzen können.

Vergleiche mit Methoden in Java.

```
Boolean phi(int x1, ..., int xk)
```

Mit $x_1, \ldots, x_k$ wird eine Ordnung der Parameter festgelegt.

Wir können dann `Boolean b = phi(3, 5, ..., 17);` benutzen.

Eine Nicht-Boolesche Anfrage

Liste alle Filme von Regisseur Georg Clooney

In der Prädikatenlogik. $\varphi(f) := \exists n\, \text{Movies}(f, \text{G-Clooney}, n)$

wobei G-Clooney ein Konstantensymbol ist, dass durch das Object "Georg Clooney" interpretiert wird.

Formal arbeiten wir also in Strukturen über der Signatur

{Actors, Movies, G-Clooney}.

In SQL.

```
SELECT Title
FROM Movies
WHERE Director="Georg Clooney"
```

Eine Nicht-Boolesche Anfrage

Antwort als Tabelle.

Answer	Good night ... und good luck

Als (unäre) Relation.

$$\varphi((\mathcal{D}, \underbrace{\text{G-Clooney}}_{\substack{\text{Konst. Symbol} \\ \text{interpret. durch} \\ \text{“Georg Clooney”}}}) = \{\text{Good night ... und good luck}\}.$$

Anmerkung. Es ist eins der eleganten Eigenschaften des relationalen Datenbankmodells, dass Antworten auf Anfragen selbst wieder Tabellen sind und somit in Datenbanken gespeichert werden können.

4.4. Semantische Folgerung und Modellklassen

Erfüllbarkeit

Analog zur Aussagenlogik können wir nun Begriffe wie Erfüllbarkeit, logische Folgerung, etc. definieren.

Definition 4.14. Sei $\varphi \in \mathsf{FO}[\sigma]$ eine Formel.

1. Eine σ-Interpretation $\mathcal{I}$ erfüllt φ, wenn $\mathcal{I}$ zu φ passt und $[\![\varphi]\!]^{\mathcal{I}} = 1$.

   Wir sagen auch: $\mathcal{I}$ ist ein Modell von φ und schreiben $\mathcal{I} \models \varphi$.

2. φ ist erfüllbar, wenn es ein Modell hat. Anderenfalls ist es unerfüllbar.

3. φ ist allgemeingültig, oder eine Tautologie, wenn alle passenden Interpretationen φ erfüllen.

Erfüllbarkeit von Mengen von Formeln

Definition 4.15. Sei $\Phi \subseteq \mathrm{FO}[\sigma]$ eine Menge von Formeln.

1. Eine Interpretation $\mathcal{I}$ passt zu Φ, wenn sie zu allen $\varphi \in \Phi$ passt.
2. Eine Interpretation $\mathcal{I}$ erfüllt Φ, wenn sie zu Φ passt und alle $\varphi \in \Phi$ erfüllt.

   Wir sagen auch: $\mathcal{I}$ ist ein Modell von Φ und schreiben $\mathcal{I} \models \Phi$.
3. Φ ist erfüllbar, wenn es ein Modell hat. Ansonsten ist es unerfüllbar.

Logische Folgerung

Definition 4.16. Sei σ eine Signatur.

Sei $\Phi \subseteq \mathsf{FO}[\sigma]$ und $\psi \in \mathsf{FO}[\sigma]$.

ψ ist eine Folgerung von Φ, geschrieben $\Phi \models \psi$, wenn für jede zu Φ und ψ passende σ-Interpretation $\mathcal{I}$ gilt:

$$\mathcal{I} \models \Phi \quad \Longrightarrow \quad \mathcal{I} \models \psi.$$

Notation. Statt $\emptyset \models \psi$ schreiben wir $\models \psi$.

Lemma 4.17.

1. Für alle $\Phi \subseteq \mathsf{FO}[\sigma]$ und $\psi \in \mathsf{FO}[\sigma]$:

$$\Phi \models \psi \quad \Longleftrightarrow \quad \Big(\Phi \cup \{\neg\psi\} \quad \text{ist unerfüllbar} \Big)$$

2. Für alle $\psi \in \mathsf{FO}[\sigma]$:

$$\models \psi \quad \Longleftrightarrow \quad \Big(\psi \text{ ist eine Tautologie} \Big).$$

Modellklassen und Axiomensysteme

Definition 4.18. Sei σ eine Signatur und $\Phi \subseteq \mathrm{FO}[\sigma]$ eine Menge von σ-Sätzen.

1. Die Modellklasse von Φ, geschrieben $\mathrm{Mod}(\Phi)$, ist die Klasse aller σ-Strukturen $\mathcal{A}$ mit $\mathcal{A} \models \Phi$.

   Falls $\Phi := \{\varphi\}$ nur einen Satz enthält, schreiben wir kurz $\mathrm{Mod}(\varphi)$.
2. Eine Klasse $\mathcal{C}$ von σ-Strukturen ist axiomatisiert durch Φ, wenn $\mathcal{C} = \mathrm{Mod}(\Phi)$.

   Wir nennen Φ ein Axiomensystem für $\mathcal{C}$.
3. Eine Klasse $\mathcal{C}$ von σ-Strukturen ist axiomatisierbar, wenn $\mathcal{C} = \mathrm{Mod}(\Phi)$ für eine Menge $\Phi \subseteq \mathrm{FO}[\sigma]$.

   Wenn es eine endliche Menge Φ mit $\mathcal{C} = \mathrm{Mod}(\Phi)$ gibt, so heißt $\mathcal{C}$ endlich axiomatisierbar oder definierbar.

Beispiel: Gruppen

Erinnerung. Eine Gruppe ist ein Tupel $(G, \circ, e, ^{-1})$, wobei $\circ$ eine zweistellige Funktion ist, e eine Konstante, und $^{-1}$ eine einstellige Funktion, so dass folgende Bedingungen erfüllt sind.

- $\circ$ ist assoziativ, d.h. $(a \circ (b \circ c)) = ((a \circ b) \circ c)$.
- e ist das neutrale Element, d.h. $a \circ e = a$.
- $^{-1}$ ist die inverse Funktion, d.h. $a \circ a^{-1} = e$.

Die Klasse aller Gruppen ist endlich axiomatisierbar durch

$$\Phi_{Gruppe} := \left\{ \begin{array}{c} \forall x \forall y \forall z \big(x \circ (y \circ z) = (x \circ y) \circ z\big) \\ \forall x (x \circ e = x) \\ \forall x (x \circ x^{-1} = e) \end{array} \right\}$$

Gilt nun $\Phi_{Gruppe} \models \psi$, für eine Formel ψ, so gilt ψ in allen Gruppen.

Beispiel: $\Phi_{Gruppe} \models \forall x (x^{-1} \circ x = e)$ (in jeder Gruppe ist das Rechtsinverse auch das Linksinverse)

Andererseits gilt $\Phi_{Gruppe} \not\models \forall x \forall y (x \circ y = y \circ x)$, da nicht alle Gruppen kommutativ sind.

Theorien

Definition 4.19. Sei σ eine Signatur.

1. Eine σ-Theorie ist eine erfüllbare Menge $T \subseteq \mathrm{FO}[\sigma]$ von Sätzen, die unter $\models$ abgeschlossen ist, d.h. wenn $T \models \psi$ für einen σ-Satz ψ gilt, dann ist $\psi \in T$.
2. Eine Theorie T ist vollständig, wenn für jeden Satz $\psi \in \mathrm{FO}[\sigma]$ gilt:

   $$\psi \in T \text{ oder } \neg\psi \in T$$

Beispiel. Wir haben bereits gesehen, dass die Klasse aller Gruppen endlich axiomatisiert ist durch

$$\Phi_{\mathit{Gruppe}} := \left\{ \begin{array}{ll} \forall x \forall y \forall z \big(x \circ (y \circ z) = (x \circ y) \circ z\big) & \\ \forall x (x \circ e = x), & \forall x (x \circ x^{-1} = e) \end{array} \right\}$$

Die Gruppentheorie ist also

$$T := \{\psi \in \mathrm{FO}[\sigma] : \psi \text{ ist ein Satz und } \Phi_{\mathit{Gruppe}} \models \psi\}.$$

Die Theorie ist nicht vollständig, da nicht alle Gruppen abelsch sind und somit $\varphi := \forall x \forall (x \circ y = y \circ x) \notin T$ und $\neg\varphi \notin T$.

Beispiel: Gruppen

Man beachte, dass ψ auch freie Variablen enthalten kann:

Beispiel: $\Phi_{Gruppe} \models x^{-1} \circ x = e$ (in jeder Gruppe ist das Rechtsinverse auch das Linksinverse)

Wir werden später einen Sequenzenkalkül für die Prädikatenlogik kennen lernen, der es erlaubt, solche Folgerungen aus einem Axiomensystem zu ziehen.

Im Prinzip könnten also alle in der Prädikatenlogik formalisierbaren Sätze der Gruppentheorie auf diese Art bewiesen werden.

Allerdings ist die Prädikatenlogik unentscheidbar und menschliche Gruppentheoretiker daher nicht überflüssig.

Weitere Beispiele

Beispiele.

1. Die Klasse aller ungerichteten Graphen ist endlich axiomatisiert durch

$$\Phi_{Graph} := \{\forall x \neg E(x,x), \quad \forall x \forall y (E(x,y) \rightarrow E(y,x))\}$$

2. Die Klasse aller linearen Ordnungen ist axiomatisiert durch die Formelmenge $\{\varphi_{ord}\}$ (siehe einige Folien vorher).

Äquivalenz und Normalformen

Äquivalenz zwischen Formeln

Definition 4.20. Sei σ eine Signatur.

Zwei σ-Formeln $\varphi, \psi \in \mathsf{FO}[\sigma]$ sind äquivalent, geschrieben $\varphi \equiv \psi$, wenn für alle σ-Interpretationen $\mathcal{I}$ passend zu φ und ψ:

$$\mathcal{I} \models \varphi \quad \Longleftrightarrow \quad \mathcal{I} \models \psi.$$

Beobachtung. Nach Definition gilt für alle Formeln $\varphi, \psi \in \mathsf{FO}[\sigma]$

$$\varphi \equiv \psi \quad \Longleftrightarrow \quad \Big(\varphi \leftrightarrow \psi \text{ ist allgemeingültig} \Big)$$

Substitution

Substitution

Genauso wie bei der Aussagenlogik wollen wir einen Begriff der Substitution einführen.

Ziel. Ersetze sinnvoll Variablen durch Terme.

Beispiel. Wenn wir in der σ_{ar}-Formel

$$\exists y\ y * y = x + x$$

die Variable x durch $(1 + 1)$ ersetzen, erhalten wir

$$\exists y\ y * y = (1 + 1)\ +\ (1 + 1).$$

Substitution

Man muss aber aufpassen, welche Variablen man ersetzt und was man für sie einsetzt!

Das Umbenennen einer Variablen in einer Formel, d.h. Substitution durch eine andere Variable, soll den Sinn der Formel nicht ändern.

Das folgende Beispiel zeigt potentielle Probleme.

Beispiel. Sei $\varphi := \exists y\ y * y = x + x$.

1. Wenn wir in φ die gebundene Variable y durch x ersetzen, erhalten wir die Formel

   $$\exists x\ x * x = x + x$$

   mit vollständig anderer Bedeutung.

   Wir sollten daher nur freie Variablen substituieren.
2. Wenn wir in φ die freie Variable x durch y ersetzen, erhalten wir

   $$\exists y\ y * y = y + y$$

   was ebenfalls eine andere Bedeutung hat.

   Wir müssen also auf Konflikte mit gebundenen Variablen achten.

Substitution

Definition 4.21. Sei σ eine Signatur.

1. Eine σ-Substitution ist eine Abbildung $\mathcal{S} : \mathrm{Dom}(\mathcal{S}) \to \mathcal{T}_\sigma$ mit endlichem Wertebereich $\mathrm{Dom}(\mathcal{S}) \subseteq \mathrm{VAR}$.

2. Für eine Substitution $\mathcal{S}$ definieren wir $\mathrm{var}(\mathcal{S})$ als die Menge der Variablen, die in einem Term im Bild der Substitution vorkommen, d.h.

$$\mathrm{var}(\mathcal{S}) := \bigcup_{x \in \mathrm{Dom}(\mathcal{S})} \mathrm{var}(\mathcal{S}(x)).$$

Substitutionen

Definition. Sei σ eine Signatur.

1. Eine σ-Substitution ist eine Abbildung $\mathcal{S} : \mathrm{Dom}(\mathcal{S}) \to \mathcal{T}_\sigma$ mit endl. Wertebereich $\mathrm{Dom}(\mathcal{S}) \subseteq \mathrm{VAR}$.

2. Für eine Substitution $\mathcal{S}$ definieren wir $\mathrm{var}(\mathcal{S})$ als die Menge der Variablen, die in einem Term im Bild der Substitution vorkommen, d.h.

$$\mathrm{var}(\mathcal{S}) := \bigcup_{x \in \mathrm{Dom}(\mathcal{S})} \mathrm{var}(\mathcal{S}(x)).$$

Beispiel. Sei $\mathcal{S}$ definiert als

$$\mathcal{S} : \begin{array}{lcl} x & \mapsto & (y + y) * 2 + z \\ y & \mapsto & z * z + v \end{array}$$

Dann $\mathrm{var}(\mathcal{S}) := \{y, z, v\}$.

Substitution in Termen

Definition 4.22. Sei $\mathcal{S}$ eine σ-Substitution.

Induktiv über die Struktur von Termen definieren wir für jeden Term $t \in \mathcal{T}_\sigma$ den Term $t\mathcal{S}$, der durch Anwendung der von $\mathcal{S}$ auf t entsteht, als:

- Wenn $t := x$, wobei $x \in \text{VAR}$, dann
$$t\mathcal{S} := \begin{cases} \mathcal{S}(x) & \text{wenn } x \in \text{Dom}(\mathcal{S}) \\ x & \text{sonst.} \end{cases}$$
- Wenn $t := c$, für ein Konstantensymbol $c \in \sigma$, dann $t\mathcal{S} := c$.
- Wenn $t := f(t_1, \ldots, t_k)$, für ein k-stelliges Funktionssymbol $f \in \sigma$ und σ-Terme $t_1, \ldots, t_k \in \mathcal{T}_\sigma$, dann $t\mathcal{S} := f(t_1\mathcal{S}, \ldots, t_k\mathcal{S})$.

Beispiel

Beispiel. Sei $\mathcal{S}$ definiert als

$$\mathcal{S}: \begin{array}{lcl} x & \mapsto & (y+z) \\ y & \mapsto & z+v \end{array}$$

Wenn $t := (x+y)$, dann

$$t\mathcal{S} := \big((y+z)+(z+v)\big)$$

Substitution in Formeln

Definition 4.23. Sei $\mathcal{S}$ eine σ-Substitution.

Induktiv über den Formelaufbau definieren wir für jede Formel $\varphi \in \mathsf{FO}[\sigma]$ die Formel $\psi\mathcal{S}$, die durch Anwenden von $\mathcal{S}$ auf φ entsteht, als:

- Wenn $\varphi := R(t_1, \ldots, t_k)$, wobei $R \in \sigma$ ein k-stellige Relationssymbol ist und $t_1, \ldots, t_k \in \mathcal{T}_\sigma$, dann $\varphi\mathcal{S} := R(t_1\mathcal{S}, \ldots, t_k\mathcal{S})$.
- Wenn $\varphi := t_1 = t_2$, für σ-Terme t_1, t_2, dann $\varphi\mathcal{S} := t_1\mathcal{S} = t_2\mathcal{S}$.
- Wenn $\varphi := \neg\psi$, für $\psi \in \mathsf{FO}[\sigma]$, dann $\varphi\mathcal{S} := \neg\, \psi\mathcal{S}$.
- Wenn $\varphi := (\psi_1 * \psi_2)$ für $\psi_1, \psi_2 \in \mathsf{FO}[\sigma]$ und $* \in \{\vee, \wedge, \rightarrow, \leftrightarrow\}$ dann $\varphi\mathcal{S} := (\psi_1\mathcal{S} * \psi_2\mathcal{S})$.

(Forts.)

Substitution in Formeln

- Wenn $\varphi := \exists x \psi$, für eine Variable x und $\psi \in \mathrm{FO}[\sigma]$, dann
  - wenn $x \notin \mathrm{var}(\mathcal{S})$ dann $\varphi\mathcal{S} := \exists x \psi \mathcal{S}'$, wobei $\mathcal{S}' := \mathcal{S}_{|\mathrm{Dom}(\mathcal{S}) \setminus \{x\}}$.

    Wenn x nicht in den Variablen von $\mathcal{S}$ vorkommt, ist die Substitution sicher und wir substituieren in ψ.

  - wenn $x \in \mathrm{var}(\mathcal{S})$ dann sei y die erste Variable in $\mathrm{VAR} := \{v_0, v_1, v_2, \dots\}$ die nicht in $\mathit{frei}(\varphi) \cup \mathrm{var}(\mathcal{S})$ vorkommt und definiere $\mathcal{S}' := \mathcal{S}_{\mathrm{Dom}(\mathcal{S}) \setminus \{x\}} \cup \{x \mapsto y\}$. Wir definieren $\varphi\mathcal{S} := \exists y \psi \mathcal{S}'$.

    Wenn x in den Variablen von $\mathcal{S}$ vorkommt, können wir nicht einfach substituieren. Wir müssen zunächst x in y umbenennen, wobei y beliebig gewählt werden kann, solange $y \notin \mathit{frei}(\varphi) \cup \mathrm{var}(\mathcal{S})$.
    Hier verwenden wir die erste unbenutzte Variable.
- Der Fall $\varphi := \forall x \psi$ ist analog.

Beispiel

Beispiel. Sei $\mathcal{S}$ wie folgt definiert

$$\mathcal{S}: \begin{array}{lcl} x & \mapsto & (y+z) \\ y & \mapsto & z+v \end{array}$$

Dann gilt $\mathrm{var}(\mathcal{S}) := \{y, z, v\}$.

Betrachte die Formel $\varphi := \exists a \forall z (x + y + z = a + x)$.

Dann

$$\begin{array}{rcl} \varphi\mathcal{S} & := & \exists a (\forall z (x + y + z = a + x)\mathcal{S} \\ & = & \exists a \, \forall v_0 (x + y + v_0 = a + x)\mathcal{S} \\ & = & \exists a \, \forall v_0 \, (x + y + v_0)\mathcal{S} = (a + x)\mathcal{S} \\ & = & \exists a \, \forall v_0 \, \big((x + z) + (z + v) + v_0 = (a + (y + z))\big) \end{array}$$

Notation

- Analog zur aussagenlog. Substitution schreiben wir für eine Substitution $\mathcal{S}$ mit $\mathrm{Dom}(\mathcal{S}) := \{x_1, \ldots, x_n\}$ und $\mathcal{S}(x_i) := t_i$, $1 \leqslant i \leqslant n$,

$$[x_1/t_1, \ldots, x_n/t_n].$$

  Insbesondere schreiben wir $\varphi[x_1/t_1, \ldots, x_n/t_n]$ statt $\varphi\mathcal{S}$.

- Wenn $\varphi(x_1, \ldots, x_k)$ eine Formel und $t_1, \ldots, t_k \in \mathcal{T}_\sigma$ Terme sind, schreiben wir

$$\varphi(t_1, \ldots, t_k) \qquad \text{statt } \varphi[x_1/t_1, \ldots, x_k/t_k].$$

  Vergleiche mit Methoden in Java.

```
Boolean phi(int x1, ..., int xk)
```

  Indem wir $x_1, \ldots, x_k$ spezifizieren, fixieren wir eine Ordnung der Parameter.

  Wir schreiben `Boolean b = phi(3, 5, ..., 17);`

Das Substitutionslemma

Lemma 4.24. Sei $\mathcal{S}$ eine σ-Substitution. Für alle σ-Formeln φ, ψ:

$$\varphi \equiv \psi \quad \Longleftrightarrow \quad \varphi\mathcal{S} \equiv \psi\mathcal{S}$$

Beweis des Substitutionslemmas

Der Beweis geht analog zum aussagenlogischen Fall.

Definition 4.25. Für jede σ-Substitution $\mathcal{S}$ und σ-Interpretation $\mathcal{I} := (\mathcal{A}, \beta)$ mit $\mathrm{var}(\mathcal{S}) \subseteq \mathrm{Dom}(\beta)$ sei

$$\mathcal{I}\mathcal{S} := (\mathcal{A}, \beta')$$

wobei $\beta' : \mathrm{Dom}(\beta) \cup \mathrm{Dom}(\mathcal{S}) \to A$ definiert ist als

$$\beta'(x) := \begin{cases} [\![\mathcal{S}(x)]\!]^{\mathcal{I}} & \text{wenn } x \in \mathrm{Dom}(\mathcal{S}) \\ \beta(x) & \text{otherwise.} \end{cases}$$

Lemma 4.26. Sei $\mathcal{S}$ eine σ-Substitution und $\mathcal{I} := (\mathcal{A}, \beta)$ eine σ-Interpretation mit $\mathrm{var}(S) \subseteq \mathrm{Dom}(\beta)$.

Für alle σ-Formeln φ mit $\mathit{frei}(\varphi) \subseteq \mathrm{Dom}(\mathcal{S}) \cup \mathrm{Dom}(\beta)$:

$$\mathcal{I} \models \varphi\mathcal{S} \quad \Longleftrightarrow \quad \mathcal{I}\mathcal{S} \models \varphi.$$

Das Ersetzungslemma

Analog zur Aussagenlogik gibt es auch ein entsprechendes Ersetzungslemma.

Lemma 4.27. Sei τ eine Signatur und seien $\varphi, \psi, \vartheta \in \mathsf{FO}[\tau]$.

Sei ϑ eine Teilformel von ψ und $\vartheta \equiv \varphi$. Ferner, sei ψ' die Formel, die aus ψ entsteht, indem ϑ durch φ ersetzt wird.

Dann gilt $\psi \equiv \psi'$.

Beispiel.

$\varphi := \neg\forall x\big(P(x) \vee (P(x) \wedge Q(x))\big)$ ist äquivalent zu $\exists x \neg P(x)$.

Also ist $\forall y P(x) \vee \neg\forall x(P(x) \vee (P(x) \wedge Q(x)))$ äquivalent zu $\forall y P(x) \vee \exists x \neg P(x)$.

Äquivalenz von Formeln

Nützliche Äquivalenzen

Lemma 4.28. Sei $\varphi, \psi \in \mathrm{FO}[\sigma]$ und $x \in \mathrm{VAR}$.

1. $\neg\exists x\ \varphi \equiv \forall x \neg\varphi$, $\qquad \neg\forall x\ \varphi \equiv \exists x \neg\varphi$
2. Wenn $x \notin \mathit{frei}(\varphi)$ dann

$$\begin{array}{lcllcl} \varphi \vee \exists x\ \psi & \equiv & \exists x(\varphi \vee \psi), & \varphi \wedge \forall x\ \psi & \equiv & \forall x(\varphi \wedge \psi) \\ \varphi \wedge \exists x\ \psi & \equiv & \exists x(\varphi \wedge \psi), & \varphi \vee \forall x\ \psi & \equiv & \forall x(\varphi \vee \psi) \end{array}$$

Die Äquivalenzen in Teil 2 folgen aus dem Koinzidenzlemma.

Lemma 4.29. Sei $\varphi \in \mathrm{FO}[\sigma]$ eine Formel und $x, y \in \mathrm{VAR}$ Variablen, so dass y nicht in φ vorkommt. Dann

$$\exists x\varphi \equiv \exists y\varphi[x/y], \qquad \forall x\varphi \equiv \forall y\varphi[x/y]$$

Beispiel

Mit den Äquivalenzen der vorherigen Folien und den schon aus der Aussagenlogik bekannten Äquivalenzen, können wir folgendes beweisen.

1.

$$\begin{aligned} & \neg\exists x\big(x = y \vee \forall y E(x,y)\big) \\ \equiv\ & \forall x \neg\big(x = y \vee \forall y E(x,y)\big) \\ \equiv\ & \forall x\big(\neg x = y \wedge \neg\forall y E(x,y)\big) \\ \equiv\ & \forall x\big(\neg x = y \wedge \exists y \neg E(x,y)\big) \end{aligned}$$

2.

$$\begin{aligned} & \big(\exists x E(x,y)\big) \vee \big(\forall z E(y,z)\big) \\ \equiv\ & \forall z\big(\exists x E(x,y)\big) \vee \big(E(y,z)\big) \\ \equiv\ & \forall z \exists x\big(E(x,y) \vee E(y,z)\big) \end{aligned}$$

Beweis der Äquivalenz

Beweis siehe Vorlesungsmitschrift.

Normalformen

Normalformen

Ähnlich wie bei der Aussagenlogik werden wir als nächstes einige syntaktische Normalformen für die Prädikatenlogik einführen, die uns das Arbeiten mit Formeln in bestimmten Situationen erleichtern.

Genauer werden wir folgende Normalformen vorstellen:

- Reduzierte Formeln
- Negationsnormalform
- Pränexnormalform

Reduzierte Formeln

Wir haben bereits gesehen, dass folgende Äquivalenzen gelten:

1. $(\psi \wedge \varphi) \quad \equiv \quad \neg(\neg\varphi \vee \neg\psi)$
2. $(\psi \leftrightarrow \varphi) \quad \equiv \quad (\varphi \rightarrow \psi) \wedge (\psi \rightarrow \varphi)$
3. $(\psi \rightarrow \varphi) \quad \equiv \quad (\neg\psi \vee \varphi)$
4. $\forall x\psi \quad \equiv \neg\exists\neg\psi$

Mit Hilfe dieser Äquivalenzen können wir also jede Formel der Prädikatenlogik in eine äquivalente Formel umwandeln, in denen die Symbole $\leftrightarrow, \rightarrow, \wedge$ nicht vorkommen.

Wir nennen solche Formeln reduzierte Formeln.

Die Negationsnormalform

Definition 4.30. Eine prädikatenlogische Formel ist in Negationsnormalform, wenn sie die Verknüpfungen $\rightarrow, \leftrightarrow$ nicht enthält und Negation nur direkt vor atomaren Formeln vorkommt.

Theorem 4.31. Jede Formel der Prädikatenlogik ist logisch äquivalent zu einer Formel in Negationsnormalform.

Beweis. Wir wissen bereits, dass die Verknüpfungen $\rightarrow, \leftrightarrow$ eliminiert werden können.

Durch wiederholte Anwendung der De Morganschen Regeln

$$\neg(\psi \wedge \varphi) \equiv (\neg\psi \vee \neg\varphi) \quad \text{und} \quad \neg(\psi \vee \varphi) \equiv (\neg\psi \wedge \neg\varphi)$$

sowie der Äquivalenzen

$$\neg\exists x\psi \equiv \forall x\neg\psi \quad \text{und} \quad \neg\forall x\psi \equiv \exists x\neg\psi$$

und $\neg\neg\psi \equiv \psi$ kann jede Formel in eine äquivalente Formel in NNF umgewandelt werden. □

Die Pränexnormalform

Als letzte Normalform stellen wir die Pränexnormalform vor. Ziel dieser Normalform ist es, die Formel so umzuschreiben, dass alle Quantoren vorne stehen.

Beispiel. $\exists u \forall x \exists y \exists z \big(R(u,u) \wedge R(x,y) \wedge \neg R(y,y) \wedge R(y,z) \big)$

Definition 4.32. Eine Formel φ ist bereinigt, wenn

- keine Variable in φ sowohl frei als auch gebunden vorkommt und
- keine Variable zweimal quantifiziert wird.

Definition 4.33. Eine Formel ψ ist in Pränexnormalform (PNF), wenn sie bereinigt ist und die Form

$$Q_1 x_1 \ldots Q_l x_l \psi(x_1, \ldots, x_l)$$

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

$Q_1, \ldots, Q_l$ heißt der (Quantoren-)Präfix von ψ.

Die Pränexnormalform

Theorem 4.34. Jede Formel der Prädikatenlogik kann effektiv in eine äquivalente Formel in Pränexnormalform übersetzt werden.

4.6. Definierbarkeit in der Prädikatenlogik

Definierbarkeit in der Prädikatenlogik

In diesem Abschnitt werden wir eine einfache Methode kennen lernen, um zu Überprüfen, ob bestimmte Eigenschaften von Strukturen in der Prädikatenlogik definierbar sind.

Beispiel. Wir betrachten eine Datenbank mit Fluginformationen.

Tabelle: Flug(Fluggesellschaft, Flugnummer, Zeit, Start, Ziel)

Wir möchten nun wissen, ob eine Möglichkeit gibt, von s nach t zu fliegen (eventuell mit Zwischenstopps).

Definierbarkeit in der Prädikatenlogik

Wir vereinfachen das Beispiel ein wenig.

Vereinfachtes Beispiel. Tabelle: Flug(Start, Ziel)

Wir möchten nun wissen, ob eine Möglichkeit gibt, von s nach t zu fliegen (eventuell mit Zwischenstopps).

Abfragen in der Prädikatenlogik. Signatur $\sigma := \{Flug, s, t\}$

s, t Konstantensymbole und Flug 2-stelliges Relationssymbol

Direktflug: $\varphi_0 := Flug(s, t)$

Ein Stopp : $\varphi_1 := \exists x \Big(Flug(s, x) \wedge Flug(x, t) \Big)$

Zwei Stopps : $\varphi_2 := \exists x_1 \exists x_2 \Big(Flug(s, x_1) \wedge Flug(x_1, x_2) \wedge Flug(x_2, t) \Big)$

Für jede feste Zahl von Stopps existiert eine Formel.

Definierbarkeit in der Prädikatenlogik

Aber gibt es eine Formel φ die genau dann in einer solchen Struktur gilt, wenn es eine Möglichkeit gibt, von s nach t zu fliegen, egal wieviele Stopps eingelegt werden müssen?

Angenommen, wir wollen zeigen, dass es keine solche gibt?

Wie kann man eine solche Aussage zeigen?

Definierbarkeit in der Prädikatenlogik

Behauptung: Es gibt keinen Satz der Prädikatenlogik, der Erreichbarkeit in diesem Sinne definiert.

Um das zu zeigen, müssen wir für jeden Satz der Prädikatenlogik zeigen, dass er Erreichbarkeit nicht definiert!

Dazu müssen wir für jeden Satz $\varphi \in \mathsf{FO}[\sigma]$ zeigen, dass es zwei σ-Strukturen $\mathcal{A}, \mathcal{B}$ gibt, so dass

1. es in $\mathcal{A}$ einen Weg von s nach t gibt, in $\mathcal{B}$ aber nicht und
2. $\mathcal{A} \models \varphi$ und $\mathcal{B} \models \varphi$ oder aber $\mathcal{A} \not\models \varphi$ und $\mathcal{B} \not\models \varphi$

Wir werden als nächstes ein Verfahren kennen lernen, dass uns diese Arbeit stark vereinfacht.

Zunächst aber einige Vorbereitungen.

Elementare Äquivalenz

Elementare Äquivalenz

Definition 4.35. Zwei σ-Strukturen $\mathcal{A}, \mathcal{B}$ sind elementar äquivalent, geschrieben $\mathcal{A} \equiv \mathcal{B}$, wenn für alle σ-Sätze ψ gilt:

$$\mathcal{A} \models \psi \iff \mathcal{B} \models \psi$$

Wenn zwei Strukturen elementar äquivalent sind, erfüllen sie also dieselben Sätze der Prädikatenlogik.

Wenn wir also zeigen wollen, dass Erreichbarkeit nicht in der Prädikatenlogik definierbar ist, würde es reichen, zwei σ-Strukturen $\mathcal{A}, \mathcal{B}$ zu finden, so dass

- es in $\mathcal{A}$ einen Weg von s nach t gibt, in $\mathcal{B}$ aber nicht und
- $\mathcal{A} \equiv \mathcal{B}$

Solche Strukturen kann es aber nicht geben.

Wir werden daher die elementare Äquivalenz noch verfeinern.

Der Quantorenrang einer Formel

Definition 4.36. Der Quantorenrang $\mathrm{qr}(\psi)$ einer Formel $\psi \in \mathrm{FO}$ ist induktiv definiert durch:

- $\mathrm{qr}(\psi) := 0$ für quantorenfreie Formeln ψ
- $\mathrm{qr}(\neg\psi) := \mathrm{qr}(\psi)$
- $\mathrm{qr}((\varphi * \psi)) := \max\{\mathrm{qr}(\varphi), \mathrm{qr}(\psi)\}$ für $* \in \{\vee, \wedge, \to, \leftrightarrow\}$
- $\mathrm{qr}(\exists x \varphi) = \mathrm{qr}(\forall x \varphi) = 1 + \mathrm{qr}(\varphi)$

Der Quantorenrang einer Formel ist also die maximale Schachtelungstiefe der Quantoren in der Formel.

Beispiel.

- $\mathrm{qr}\Big(\exists x \forall y \big(x = y \vee R(x, y, z)\big)\Big) = 2$
- $\mathrm{qr}\Big(\exists x \big(T(x) \vee \forall y\, R(x, y, z)\big)\Big) = 2$
- $\mathrm{qr}\Big(\exists x\, T(x) \vee \forall y \big(R(y, y, z) \to y = z\big)\Big) = 1$

m-Äquivalenz

Definition 4.37. Sei $m \in \mathbb{N}$. Zwei σ-Strukturen $\mathcal{A}, \mathcal{B}$ sind *m-äquivalent*, geschrieben $\mathcal{A} \equiv_m \mathcal{B}$, wenn für alle σ-Sätze ψ mit Quantorenrang $\mathrm{qr}(\psi) \leqslant m$ gilt:

$$\mathcal{A} \models \psi \iff \mathcal{B} \models \psi$$

Wenn zwei Strukturen elementar äquivalent sind, erfüllen sie also dieselben Sätze der Prädikatenlogik bis zum Quantorenrang m.

m-Äquivalenz erlaubt eine feinere Unterscheidung zwischen Strukturen.

Offenbar sind Strukturen $\mathcal{A}, \mathcal{B}$ die elementar äquivalent sind auch m-äquivalent für alle m.

m-Äquivalenz

Wir erweitern die Begriffe der Äquivalenz und m-Äquivalenz noch auf Strukturen mit ausgezeichneten Elementen.

Definition 4.38. Seien $\mathcal{A}, \mathcal{B}$ σ-Strukturen und $\overline{a} \in A^k, \overline{b} \in B^k$ k-Tupel von Elementen.

1. $(\mathcal{A}, \overline{a})$ und $(\mathcal{B}, \overline{b})$ sind m-äquivalent, geschrieben $(\mathcal{A}, \overline{a}) \equiv_m (\mathcal{B}, \overline{b})$, wenn für alle σ-Formeln $\psi(\overline{x})$ mit Quantorenrang $\mathrm{qr}(\psi) \leqslant m$ und freien Variablen $\overline{x} := x_1, \ldots, x_k$ gilt:

$$\mathcal{A} \models \psi[\overline{a}] \iff \mathcal{B} \models \psi[\overline{b}]$$

2. $(\mathcal{A}, \overline{a})$ und $(\mathcal{B}, \overline{b})$ sind elementar äquivalent, geschrieben $(\mathcal{A}, \overline{a}) \equiv (\mathcal{B}, \overline{b})$, wenn für alle σ-Formeln $\psi(\overline{x})$ und freien Variablen $\overline{x} := x_1, \ldots, x_k$ gilt:

$$\mathcal{A} \models \psi[\overline{a}] \iff \mathcal{B} \models \psi[\overline{b}]$$

m-Äquivalenz

Erinnerung: Sei $m \in \mathbb{N}$. Zwei σ-Strukturen $\mathcal{A}, \mathcal{B}$ sind m-äquivalent, geschrieben $\mathcal{A} \equiv_m \mathcal{B}$, wenn für alle σ-Sätze ψ mit Quantorenrang $\mathrm{qr}(\psi) \leqslant m$ gilt:

$$\mathcal{A} \models \psi \iff \mathcal{B} \models \psi$$

Definierbarkeit in der Prädikatenlogik. m-Äquivalenz eignet sich besser als elementare Äquivalenz zum Beweis der Nicht-Definierbarkeit bestimmter Aussagen.

Wenn wir zeigen wollen, dass Erreichbarkeit nicht in der Prädikatenlogik definierbar ist, reicht, für alle m zwei σ-Strukturen $\mathcal{A}_m, \mathcal{B}_m$ zu finden, so dass

- es in $\mathcal{A}_m$ einen Weg von s nach t gibt, in $\mathcal{B}_m$ aber nicht und
- $\mathcal{A}_m \equiv_m \mathcal{B}_m$

m-Äquivalenz und Definierbarkeit

Allgemeiner können wir folgendes zeigen:

Lemma 4.39. Sei σ eine Signatur und $\mathcal{C}$ eine Klasse von σ-Strukturen.

Falls es für alle $m \geqslant 1$ zwei σ-Strukturen $\mathcal{A}_m, \mathcal{B}_m$ zu finden, so dass

- $\mathcal{A}_m \in \mathcal{C}$ aber $\mathcal{B}_m \notin \mathcal{C}$
- $\mathcal{A}_m \equiv_m \mathcal{B}_m$

dann gibt es keinen Satz $\varphi \in \mathsf{FO}[\sigma]$ der $\mathcal{C}$ definiert, d.h. so dass $\mathsf{Mod}(\varphi) = \mathcal{C}$.

Beweis. Angenommen, es gäbe ein $\varphi \in \mathsf{FO}[\sigma]$ mit $\mathcal{C} = \mathsf{Mod}(\varphi)$. Sei $m := \mathrm{qr}(\varphi)$. Dann ist aber $\mathcal{A}_m \in \mathcal{C}$ und somit $\mathcal{A}_m \models \varphi$.

Da aber $\mathcal{A}_m \equiv_m \mathcal{B}_m$ gilt auch $\mathcal{B}_m \models \varphi$, im Widerspruch zu $\mathcal{B}_m \notin \mathcal{C}$. $\square$

Ehrenfeucht-Fraïssé Spiele

Partielle Isomorphismen

Wir werden jetzt eine spieltheoretische Methode kennen lernen, um elementare oder m-Äquivalenz zwischen Strukturen testen zu können. Dazu benötigen wir zunächst ein wenig Notation.

Vereinbarung. Zur Vereinfachung der Notation betrachten wir in diesem Abschnitt nur relationale Signaturen σ, d.h. Signaturen, in denen nur Relationssymbole vorkommen.

Definition 4.40. Sei σ eine (relationale) Signatur. Ein partieller Isomorphismus zwischen zwei σ-Strukturen $\mathcal{A}, \mathcal{B}$ ist eine injektive Abbildung $h : A' \to B$, wobei $A' \subseteq A$, so dass für alle

- $k \geqslant 0$ und alle
- k-stelligen Relationssymbole $R \in \sigma \cup \{=\}$ und alle
- $a_1, ..., a_k \in A'$

gilt:

$$(a_1, \ldots, a_k) \in R^{\mathcal{A}} \quad \text{gdw.} \quad (h(a_1), ..., h(a_k)) \in R^{\mathcal{B}}$$

Partielle Isomorphismen

Definition 4.41. Sei σ eine (relationale) Signatur. Ein partieller Isomorphismus zwischen zwei σ-Strukturen $\mathcal{A}, \mathcal{B}$ ist eine injektive Abbildung $h : A' \to B$, wobei $A' \subseteq A$, so dass für alle

- $k \geqslant 0$ und alle
- k-stelligen Relationssymbole $R \in \sigma \cup \{=\}$ und alle
- $a_1, ..., a_k \in A'$

gilt:

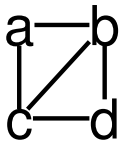
$$(a_1, \ldots, a_k) \in R^{\mathcal{A}} \quad \text{gdw.} \quad (h(a_1), ..., h(a_k)) \in R^{\mathcal{B}}$$

Beobachtungen.

- Achtung: ein partieller Isomorphismus sagt nur was über Teile der Strukturen aus, abers nichts über die Strukturen insgesamt!
- Die leere Abbildung, d.h. die Abbildung mit Definitionsbereich $\emptyset$ ist ein partieller Isomorphismus.
- Ein nicht-leerer partieller Isomorphismus von $\mathcal{A}$ nach $\mathcal{B}$ mit Definitionsbereich $A' \subseteq A$ und Bildbereich $B' \subseteq B$ ist ein Isomorphismus zwischen den von A und B induzierten Substrukturen von $\mathcal{A}$ und $\mathcal{B}$.

Beispiele

Beispiel. Sei $\sigma := \{E\}$ und seien $\mathcal{A}, \mathcal{B}$ wie folgt gegeben:

$\mathcal{A}$: 1—2—3—4—5 und $\mathcal{B}$: 

Dann ist $\pi : 2 \mapsto a, 3 \mapsto b, 4 \mapsto d$ ein partieller Isomorphismus zwischen $\mathcal{A}$ und $\mathcal{B}$.

Dann ist $\pi : 2 \mapsto a, 3 \mapsto b, 4 \mapsto c$ ist jedoch kein partieller Isomorphismus zwischen $\mathcal{A}$ und $\mathcal{B}$.

Partielle Isomorphismen

Proposition 4.42. Sei σ eine relationale Signatur, $\mathcal{A}, \mathcal{B}$ σ-Strukturen und $\overline{a} := a_1, \ldots, a_k \in A^k$ und $\overline{b} := b_1, \ldots, b_k \in B^k$. Dann sind folgende Aussagen äquivalent:

1. Die Abbildung $h : \{a_1, \ldots, a_k\} \to \{b_1, \ldots, b_k\}$ definiert durch $h(a_i) := b_i$ für alle $1 \leqslant i \leqslant k$ ist ein partieller Isomorphismus.
2. Für alle atomaren Formeln $\psi(x_1, \ldots, x_k)$ mit $\mathit{frei}(\psi) \subseteq \{x_1, \ldots, x_k\}$ gilt:
$$\mathcal{A} \models \psi[\overline{a}] \quad \text{gdw.} \quad \mathcal{B} \models \psi[\overline{b}]$$
3. Für alle quantorenfreien Formeln $\psi(x_1, \ldots, x_k)$ mit $\mathit{frei}(\psi) \subseteq \{x_1, \ldots, x_k\}$ gilt:
$$\mathcal{A} \models \psi[\overline{a}] \quad \text{gdw.} \quad \mathcal{B} \models \psi[\overline{b}]$$

Die Existenz eines partiellen Isomorphismus' beschreibt also $\equiv_0$ zwischen zwei Strukturen, sagt aber nichts über $\equiv_m$ mit $m > 0$ aus.

Ehrenfeucht-Fraïssé Spiele

Sei σ eine Signatur und seien $m, k \in \mathbb{N}$. Seien ferner $\mathcal{A}$ und $\mathcal{B}$ zwei σ-Strukturen und $\overline{a} := a_1, \ldots, a_k \in A^k$ und $\overline{b} := b_1, \ldots, b_k \in B^k$.

Spieler und deren Ziele. Das m-Runden Ehrenfeucht-Fraïssé Spiel $\mathfrak{G}_m(\mathcal{A}, \overline{a}, \mathcal{B}, \overline{b})$ wird von zwei Spielern, dem Herausforderer und der Duplikatorin, gespielt.

Herausforderers Ziel: Zeige, dass $(\mathcal{A}, \overline{a}) \not\equiv_m (\mathcal{B}, \overline{b})$

Duplikatorins Ziel: Zeige, dass $(\mathcal{A}, \overline{a}) \equiv_m (\mathcal{B}, \overline{b})$

Notation. Ist $k = 0$ so schreiben wir kurz $\mathfrak{G}_m(\mathcal{A}, \mathcal{B})$

Ehrenfeucht-Fraïssé Spiele

Sei σ eine Signatur und seien $m, k \in \mathbb{N}$. Seien ferner $\mathcal{A}$ und $\mathcal{B}$ zwei σ-Strukturen und $\overline{a} := a_1, \ldots, a_k \in A^k$ und $\overline{b} := b_1, \ldots, b_k \in B^k$.

Die Regeln des Spiels. Eine Partie des Spiels besteht aus m Runden.

In Runde $i = 1, ..., m$:

- Zunächst wählt der Herausforderer entweder ein Element $a'_i \in A$ oder $b'_i \in B$.
- Danach antwortet die Duplizierin. Hat der Herausforder ein $a'_i \in A$ gewählt, wählt nun die Duplizierin ein $b'_i \in B$. Anderenfalls wählt sie $a'_i \in A$.

Nach Runde m wird der Gewinner ermittelt:

Die Duplizierin hat gewonnen, wenn die Abbildung

$$h : a_1 \mapsto b_1, \ldots, a_k \mapsto b_k, a'_1 \mapsto b'_1, \ldots, a'_m \mapsto b'_m$$

ein partieller Isomorphismus von $\mathcal{A}$ nach $\mathcal{B}$ ist.

Ehrenfeucht-Fraïssé Spiele

Sei σ eine Signatur und seien $m, k \in \mathbb{N}$. Seien ferner $\mathcal{A}$ und $\mathcal{B}$ zwei σ-Strukturen und $\overline{a} := a_1, \ldots, a_k \in A^k$ und $\overline{b} := b_1, \ldots, b_k \in B^k$.

Gewinnstrategien. Eine Gewinnstrategie für den Herausforderer im Spiel $\mathfrak{G}_m(\mathcal{A}, \overline{a}, \mathcal{B}, \overline{b})$ ist eine Funktion, die ihm in jeder erreichbaren Spielposition einen möglichen Zug angibt, mit dem er die Partie gewinnt, egal wie die Duplikatorin spielt.

Analog sind Gewinnstrategien für die Duplikatorin definiert.

Wir sagen: Der Herausforderer gewinnt das Spiel $\mathfrak{G}_m(\mathcal{A}, \overline{a}, \mathcal{B}, \overline{b})$ (anstatt einer einzelnen Partie), wenn er eine Gewinnstrategie für das Spiel hat.

Ansonsten gewinnt die Duplikatorin.

Beispiel

Beispiel Sei $\sigma := \{<\}$, mit $<$ einem 2-stelligem Relationssymbol. Sei $m > 0$.

Seien $\mathcal{A} := (A, <^{\mathcal{A}}), \mathcal{B} := (B, <^{\mathcal{B}})$ zwei endliche lineare Ordnungen.

Wenn $|A| = |B|$ oder aber $|A|, |B| > 2^m$, so gewinnt die Duplikatorin das Spiel $\mathfrak{G}_m(\mathcal{A}, \mathcal{B})$.

Ehrenfeucht-Fraïssé Spiele

Eine wichtige Variante des m-Runden Ehrenfeucht-Fraïssé Spiels ist das Spiel $\mathfrak{G}(\mathcal{A}, \overline{a}, \mathcal{B}, \overline{b})$ mit unbeschränkter Zugzahl.

Hier wählt der Herausforderer zunächst eine Zahl $m \geqslant 0$ und dann wird das m-Runden Spiel $\mathfrak{G}_m(\mathcal{A}, \overline{a}, \mathcal{B}, \overline{b})$ gespielt.

Theorem 4.43. (Satz von Ehrenfeucht)

Sei σ eine Signatur und sei $k \in \mathbb{N}$. Seien ferner $\mathcal{A}$ und $\mathcal{B}$ zwei σ-Strukturen und $\overline{a} := a_1, \ldots, a_k \in A^k$ und $\overline{b} := b_1, \ldots, b_k \in B^k$.

1. Folgende Aussagen sind äquivalent:
   - 1.1 $(\mathcal{A}, \overline{a}) \equiv (\mathcal{B}, \overline{b})$
   - 1.2 Die Duplikatorin gewinnt $\mathfrak{G}(\mathcal{A}, \overline{a}, \mathcal{B}, \overline{b})$
2. Für alle $m \geqslant 0$ sind folgende Aussagen sind äquivalent:
   - 2.1 $(\mathcal{A}, \overline{a}) \equiv_m (\mathcal{B}, \overline{b})$
   - 2.2 Die Duplikatorin gewinnt $\mathfrak{G}_m(\mathcal{A}, \overline{a}, \mathcal{B}, \overline{b})$

m-Isomorphietypen

Als Hilfsmittel zum Beweis des Satzes von Ehrenfeucht verwenden wir folgende induktiv definierte Formeln $\varphi^m_{\mathcal{A},\overline{a}}(\overline{x})$:

m-Isomorphietypen oder Hintikka-Formeln.

Sei σ eine Signatur und sei $k \in \mathbb{N}$. Sei ferner $\mathcal{A}$ eine σ-Struktur, $\overline{a} := a_1, \ldots, a_k \in A^k$ und $\overline{x} := x_1, \ldots, x_k$.

$$\varphi^0_{\mathcal{A},\overline{a}}(\overline{x}) := \bigwedge \left\{ \psi(\overline{x}) : \psi \text{ ist atomar oder negiert atomar und } \mathcal{A} \models \psi[\overline{a}] \right\}$$

und für $m \geqslant 0$

$$\varphi^{m+1}_{\mathcal{A},\overline{a}}(\overline{x}) := \bigwedge_{a \in A} \exists x_{k+1} \varphi^{m-1}_{\mathcal{A},\overline{a},a}((\overline{x}, x_{k+1}) \wedge \forall x_{k+1} \bigvee_{a \in A} \varphi^{m-1}_{\mathcal{A},\overline{a},a}(\overline{x}, x_{k+1})$$

Ehrenfeucht-Fraïssé Spiele

Lemma 4.44.

Sei σ eine Signatur und sei $k \in \mathbb{N}$. Seien ferner $\mathcal{A}$ und $\mathcal{B}$ zwei σ-Strukturen und $\overline{a} := a_1, \ldots, a_k \in A^k$ und $\overline{b} := b_1, \ldots, b_k \in B^k$.

Für alle $m \geqslant 0$ sind folgende Aussagen sind äquivalent:

1. Die Duplikatorin gewinnt $\mathfrak{G}_m(\mathcal{A}, \overline{a}, \mathcal{B}, \overline{b})$
2. $\mathcal{B} \models \varphi^m_{\mathcal{A},\overline{a}}[\overline{b}]$
3. $(\mathcal{A}, \overline{a}) \equiv_m (\mathcal{B}, \overline{b})$

Anwendung

Definition 4.45. Sei σ eine relationale Signatur und $\mathcal{K}$ eine Klasse von σ-Strukturen.

Eine Klasse $\mathcal{C} \subseteq \mathcal{K}$ ist in $\mathcal{K}$ FO-definierbar, wenn es einen Satz $\psi \in \mathrm{FO}[\sigma]$ gibt, so dass

$$\mathcal{C} \quad = \quad \{\mathcal{A} \in \mathcal{K} : \mathfrak{A} \models \psi\}$$

Korollar 4.46. Sei σ eine relationale Signatur, $\mathcal{K}$ eine Klasse von σ-Strukturen und $\mathcal{C} \subseteq \mathcal{K}$. Dann sind äquivalent:

1. $\mathcal{C}$ ist nicht in $\mathcal{K}$ FO-definierbar.
2. Für alle $m \in \mathbb{N}$ gibt es $\mathcal{A}_m \in \mathcal{C}$ und $\mathcal{B}_m \in \mathcal{K} \setminus \mathcal{C}$ mit $\mathcal{A}_m \equiv_m \mathcal{B}_m$.
3. Für alle $m \in \mathbb{N}$ gibt es $\mathcal{A}_m \in \mathcal{C}$ und $\mathcal{B}_m \in \mathcal{K} \setminus \mathcal{C}$, so dass die Duplikatorin das Spiel $\mathfrak{G}_m(\mathcal{A}_m, \mathcal{B}_m)$ gewinnt.

Anwendung

Theorem. 4.47. Sei $\sigma := \{<\}$ und $\mathcal{K}$ die Klasse aller endlichen linearen Ordnungen. Die Klasse $\mathcal{C} \subseteq \mathcal{K}$ der endlichen linearen Ordnungen gerader Länge ist in $\mathcal{K}$ nicht FO-definierbar.

Beweis. Wir haben schon gesehen, dass wenn $\mathcal{A} := (A, <^{\mathcal{A}}), \mathcal{B} := (B, <^{\mathcal{B}})$ zwei endliche lineare Ordnungen sind, so gewinnt die Duplikatorin das Spiel $\mathfrak{G}_m(\mathcal{A}, \mathcal{B})$ genau dann, wenn $|A| = |B|$ oder aber $|A|, |B| > 2^m$.

Wir brauchen also für $m \geqslant 0$ nur $\mathcal{A}_m$ als lineare Ordnung der Länge 2^{m+1} und $\mathcal{B}_m$ als lineare Ordnung der Länge $2^{m+1} + 1$ zu wählen.

Dann gilt $\mathcal{A}_m \equiv_m \mathcal{B}_m$ aber $\mathcal{A}_m \in \mathcal{C}$ und $\mathcal{B}_m \in \mathcal{K} \setminus \mathcal{C}$. □

Erreichbarkeit

Betrachten wir noch einmal das Beispiel vom Anfang des Abschnitts.

Sei $\sigma := \{E, S, T\}$, wobei E ein 2-stelliges Relationssymbol ist und S, T einstellige Relationssymbole sind.

Sei $\mathcal{K}$ die Klasse aller σ-Strukturen, in denen S, T durch Relationen interpretiert werden, die nur ein einziges Element enthalten.

Sei $\mathit{Reach} \subseteq \mathcal{K}$ die Klasse aller σ-Strukturen, in denen es einen Pfad vom Element in S zum Element in T gibt.

Behauptung. *Reach* ist nicht in $\mathcal{K}$ FO-definierbar.

4.8. Der Sequenzenkalkül

Erinnerung: Logische Schlüsse

Frage. Wie können wir logisches Schließen formalisieren?

- Gegeben eine Menge von Voraussetzungen:

  $\{$ “Zug zu spät und keine Taxis impliziert Peter war spät", “Peter war nicht zu spät”, “Zug war zu spät” $\}$

  können wir daraus herleiten, dass “es gab Taxis”?
- Gibt es allgemeine Verfahren, um solche Schlüsse aus einer Menge von Voraussetzungen zu ziehen? Eine Methode,
  - mit der nur korrekte Schlüsse gezogen werden können
  - die allgemein genug ist, alle gültigen Schlüsse ziehen zu können?

Eine solche Methode ist die Resolution.

Wir werden jetzt eine weitere Methode kennen lernen, den Sequenzenkalkül.

Logische Ableitungen

Wir werden ein Beweissystem kennen lernen, den Sequenzenkalkül.

Der Sequenzenkalkül formalisiert die Art, in der Mathematiker oder Logiker aus Aussagen Folgerungen ziehen.

Wir führen dazu zunächst den Begriff der Sequenz ein, d.h. Aussagen der Form

$$\underbrace{\left\{\begin{array}{c}\textit{“Zug spät und keine Taxis impliziert Peter spät”},\\ \textit{“Zug zu spät”},\\ \textit{“Peter nicht zu spät”}\end{array}\right\}}_{\textit{Voraussetzung}} \Rightarrow \underbrace{\textit{“es gab Taxis”}}_{\textit{Konklusion}}$$

Die Bedeutung einer Sequenz ist, dass aus den Voraussetzungen die Konklusion geschlossen werden kann.

Logische Ableitungen

Wir werden auch unendliche Mengen von Voraussetzungen zulassen, interessieren uns aber meistens nur für eine Konklusion.

In der Mitte eines Beweises ist es aber oft nützlich, verschiedene Möglichkeiten zu betrachten, also Hypothesen aufzustellen. D.h. wir werden zwischenzeitlich mehr als eine Folgerung betrachten.

Zum Beispiel könnten wir in dem Zugbeispiel als Zwischenschritt annehmen

$$\underbrace{\left\{\begin{array}{c}\text{“Zug spät und keine Taxis impliziert Peter spät”},\\ \text{“Zug zu spät”},\\ \text{“Peter nicht zu spät”}\end{array}\right\}}_{\textit{Voraussetzung}} \Rightarrow \underbrace{\begin{array}{c}\text{“es gab Taxis”}\\ \text{"Peter spät"}\end{array}}_{\textit{Konklusion}}$$

Die Bedeutung ist, dass wenn die Voraussetzungen stimmen, stimmt mindestens eine Konklusion.

Wir können dann aus der Voraussetzung “Peter nicht spät” die eigentliche Folgerung ableiten.

Ein Sequenzenkalkül

Wir stellen zunächst einen Sequenzenkalkül für die Aussagenlogik vor und erweitern ihn später auf die Prädikatenlogik.

- Es gibt viele verschiedene Sequenzenkalküle, abhängig von den in der Logik verwendeten Verknüpfungen.
- Wir werden hier einen Kalkül für Formeln ohne $\leftrightarrow$ vorstellen.
- Wie bereits gezeigt, ist dies keine Einschränkung der Allgemeinheit.

Ein Sequenzenkakül

Notation 4.48.

1. Im Folgenden benutzen wir Φ, Δ für Mengen von Formeln. Dabei wird Δ immer endlich sein, Φ kann endlich oder unendlich sein.
2. Wenn Φ und Δ endlich sind, schreiben wir
   2.1 $\bigwedge \Phi$ für die Konjunktion $\varphi_1 \wedge \cdots \wedge \varphi_n$ aller Formeln in Φ.
   2.2 $\bigvee \Delta$ für die Disjunktion $\psi_1 \vee \cdots \vee \psi_m$ aller Formeln in Δ.

Definition 4.49.

1. Eine Sequenz ist eine Aussage der Form

$$\Phi \Rightarrow \Delta.$$

   Wir nennen Φ die Voraussetzungen und Δ die Konklusionen. (Bisweilen auch Antezedenz und Sukzedenz)
2. Eine Sequenz $\Phi \Rightarrow \Delta$ ist gültig, wenn jede Belegung β die alle Formeln in Φ erfüllt mindestens eine Formel in Δ erfüllt.
3. Ist $\Phi \Rightarrow \Delta$ nicht gültig, so gibt es eine Belegung β die alle Formeln in Φ erfüllt, aber keine in Δ. Wir sagen: β falsifiziert die Sequenz.

Axiome und Theoreme

Beispiel. Die folgende Sequenz ist gültig

$$\{(T \wedge \neg C) \rightarrow L, \quad \neg L, \quad T\} \quad \Rightarrow \quad \{C\}$$

Beispiel 4.50.

1. Jede Sequenz $\Phi \Rightarrow \Delta$ mit $\Phi \cap \Delta \neq \emptyset$ ist gültig.
2. Eine Sequenz $\Phi \Rightarrow \emptyset$ ist gültig gdw. Φ unerfüllbar ist.
3. Eine Sequenz $\emptyset \Rightarrow \Delta$ (mit Δ endl.) ist gültig gdw. $\bigvee \Delta$ gültig ist.

Definition 4.51.

1. Ein Axiom des Sequenzenkalküls ist eine Sequenz $\Phi \Rightarrow \Delta$, so dass $\Phi \cap \Delta \neq \emptyset$.
2. Ein Theorem ist eine gültige Sequenz $\emptyset \Rightarrow \{\psi\}$, d.h. eine Sequenz ohne Voraussetzungen mit genau einer Konklusion.

Ein Sequenzenkalkül

Das Ziel des Sequenzenkalküls ist das Ableiten gültiger Sequenzen aus anderen gültigen Sequenzen durch Regeln der Form

$$\frac{\Phi \Rightarrow \Delta}{\Phi' \Rightarrow \Delta'}$$

Die Idee ist, dass wenn man einmal gezeigt hat, dass $\Phi \Rightarrow \Delta$ gültig ist, dann stellt die Regel sicher, dass auch $\Phi' \Rightarrow \Delta'$ gültig ist.

Wir nennen $\Phi \Rightarrow \Delta$ die Prämisse der Regel und $\Phi' \Rightarrow \Delta'$ ihre Konsequenz.

Beispiel.

$$\frac{\{p, q\} \Rightarrow \{q\}}{\{p \wedge q\} \Rightarrow \{q\}}$$

Beispiel

Beispiel.

$$\frac{\{p, q\} \Rightarrow \{q\}}{\{p \wedge q\} \Rightarrow \{q\}}$$

Notation.

1. Wir schreiben Mengen $\{\varphi_1, \ldots, \varphi_n\}$ als $\varphi_1, \ldots, \varphi_n$
2. Wenn Φ eine Menge von Formeln ist und ψ eine Formel, dann schreiben wir Φ, ψ für $\Phi \cup \{\psi\}$.

Die obige Regel wird also wie folgt notiert:

$$\frac{p, q \Rightarrow q}{p \wedge q \Rightarrow q}$$

Die Prämisse ist $p, q \Rightarrow q$ und die Konsequenz $p \wedge q \Rightarrow q$.

Die Regeln für die Konjunktion

Es gibt zwei Regeln für die Konjunktion, eine um sie in der Prämisse einzuführen und eine für die Konklusion.

Konjunktion auf der linken Seite.

$$(\wedge \Rightarrow)\ \frac{\Phi, \psi, \varphi \Rightarrow \Delta}{\Phi, \psi \wedge \varphi \Rightarrow \Delta}$$

Beispiel. Die Regel erlaubt Beweise wie den folgenden

$$(\wedge \Rightarrow)\ \frac{p, q \Rightarrow q}{p \wedge q \Rightarrow q}$$

Die Regeln für die Konjunktion

Die folgende Regel kann benutzt werden, um Konjunktionen auf der rechten Seite einzuführen. Sie hat zwei Prämissen.

Konjunktion auf der rechten Seite.

$$(\Rightarrow \wedge)\ \frac{\Phi \Rightarrow \Delta, \psi \qquad \Phi \Rightarrow \Delta, \varphi}{\Phi \Rightarrow \Delta, \psi \wedge \varphi}$$

Beispiel. Die Regel erlaubt Beweise wie den folgenden

$$(\Rightarrow \wedge)\ \frac{p, q, r \Rightarrow q \qquad p, q, r \Rightarrow r}{p, q, r \Rightarrow q \wedge r}$$

Beweise im Sequenzenkalkül

Wir können Regeln zu komplizierteren Beweisen kombinieren.

Beispiel.

$$\dfrac{\dfrac{p, q, r \Rightarrow q \quad p, q, r \Rightarrow r}{p, q, r \Rightarrow q \wedge r}}{p \wedge q, r \Rightarrow q \wedge r}$$

Aus diesem Beweis können wir folgende Aussage ablesen: wenn

- die Sequenz $p, q, r \Rightarrow q$ gültig ist und
- die Sequenz $p, q, r \Rightarrow r$ gültig ist

dann ist auch $p \wedge q, r \Rightarrow q \wedge r$ gültig. (wird später bewiesen)

Wir wissen bereits, dass $p, q, r \Rightarrow q$ und $p, q, r \Rightarrow r$ gültig sind, da sie Axiome sind.

Also können wir folgern, dass $p \wedge q, r \Rightarrow q \wedge r$ gültig ist.

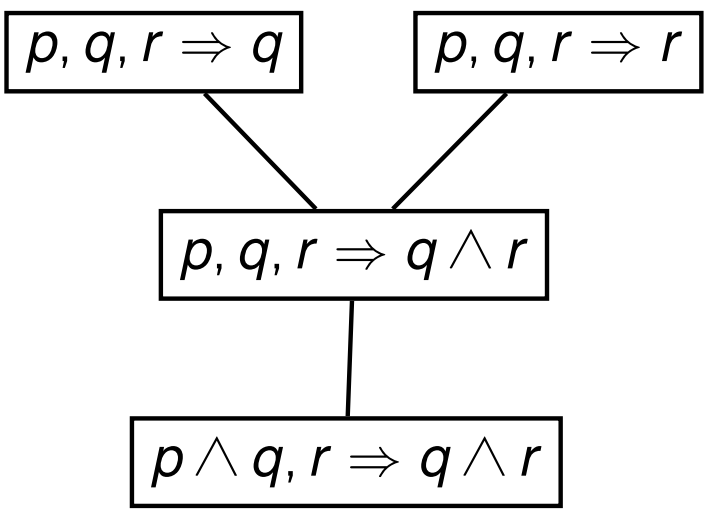
Beweise im Sequenzenkalkül

Definition 4.52. Ein Beweis im Sequenzenkalkül ist ein Baum, dessen Knoten wie folgt mit Sequenzen beschriftet sind.

- Die Blätter sind mit Axiomen beschriftet.
- Jeder innere Knoten ist mit der Konsequenz einer Regel beschriftet. Die Kinder des Knotens sind mit den Prämissen der Regel beschriftet. Jeder innere Knoten hat also entweder ein oder zwei Kinder.

Beispiel.

$$\dfrac{\dfrac{\overline{p, q, r \Rightarrow q} \quad \overline{p, q, r \Rightarrow r}}{p, q, r \Rightarrow q \wedge r}}{p \wedge q, r \Rightarrow q \wedge r}$$



Notation. Wir "überstreichen" Axiome um das Ende des Astes zu markieren.

Beweise im Sequenzenkalkül

Definition 4.53. Eine Sequenz $\Phi \Rightarrow \Delta$ ist beweisbar, oder kann abgeleitet werden, im Sequenzenkalkül, wenn sie als Beschriftung eines Knotens in einem Beweis vorkommt.

Beispiel. Die Sequenz $p \wedge q, r \Rightarrow q \wedge r$ ist im Sequenzenkalkül beweisbar.

Wir werden später beweisen, dass alle ableitbaren Sequenzen gültig sind.

Der vorherige Beweis zeigt also, dass $p \wedge q, r \Rightarrow q \wedge r$ gültig ist und somit ist

$$(p \wedge q) \wedge r \rightarrow (q \wedge r)$$

eine Tautologie.

In dieser Art wird der Sequenzenkalkül zum Beweis logisch korrekter Schlüsse benutzt.

Die Regeln für die Negation

Es gibt zwei Regeln für die Negation, eine für die linke und eine für die rechte Seite.

Regeln der Negation.

$$(\neg \Rightarrow)\ \frac{\Phi \Rightarrow \Delta, \psi}{\Phi, \neg\psi \Rightarrow \Delta} \qquad (\Rightarrow \neg)\ \frac{\Phi, \psi \Rightarrow \Delta}{\Phi \Rightarrow \Delta, \neg\psi}$$

Beispiel. Die Regeln erlauben es Beweise wie den folgenden zu bilden

$$\dfrac{\dfrac{\overline{\varphi \Rightarrow \varphi}}{\Rightarrow \neg\varphi, \varphi}}{\neg\neg\varphi \Rightarrow \varphi} \qquad \dfrac{\dfrac{\overline{\varphi \Rightarrow \varphi}}{\neg\varphi, \varphi \Rightarrow}}{\varphi \Rightarrow \neg\neg\varphi}$$

Die Regeln der Disjunktion

Es gibt zwei Regeln für die Disjunktion, eine für die linke und eine für die rechte Seite.

Disjunktion auf der rechten Seite.

$$(\Rightarrow \vee)\ \frac{\Phi \Rightarrow \Delta, \varphi, \psi}{\Phi \Rightarrow \Delta, \varphi \vee \psi}$$

Beispiel. Die Regel erlaubt Beweise wie den folgenden

$$\cfrac{\cfrac{\overline{\varphi \Rightarrow \varphi}}{\Rightarrow \varphi, \neg\varphi}}{\Rightarrow \neg\varphi \vee \varphi}$$

Das Prinzip $\neg\varphi \vee \varphi$ ist eine der wichtigsten Prinzipien der Logik und wird tertium non datur genannt. (bisweilen “law of excluded middle”)

Die Regeln der Disjunktion

Es gibt zwei Regeln für die Disjunktion, eine für die linke und eine für die rechte Seite.

Disjunktion auf der linken Seite.

$$(\vee \Rightarrow)\ \frac{\Phi, \varphi \Rightarrow \Delta \qquad \Phi, \psi \Rightarrow \Delta}{\Phi, \varphi \vee \psi \Rightarrow \Delta}$$

Beispiel. Die Regel erlaubt Beweise wie den folgenden

$$\dfrac{\dfrac{\dfrac{}{\varphi \Rightarrow \psi, \varphi}}{\varphi \Rightarrow \psi \vee \varphi} \qquad \dfrac{\dfrac{}{\psi \Rightarrow \psi, \varphi}}{\psi \Rightarrow \psi \vee \varphi}}{\varphi \vee \psi \Rightarrow \psi \vee \varphi}$$

Regeln für die Implikation

Es gibt zwei Regeln für die Implikation, eine für die linke und eine für die rechte Seite.

Implikation auf der linken Seite.

$$(\rightarrow\Rightarrow)\ \frac{\Phi \Rightarrow \Delta, \varphi \qquad \Phi, \psi \Rightarrow \Delta}{\Phi, \varphi \rightarrow \psi \Rightarrow \Delta}$$

Implikation auf der rechten Seite.

$$(\Rightarrow\rightarrow)\ \frac{\Phi, \varphi \Rightarrow \Delta, \psi}{\Phi \Rightarrow \Delta, \varphi \rightarrow \psi}$$

Die Regeln des aussagenlogischen Sequenzenkalküls

$$(\neg \Rightarrow)\ \frac{\Phi \Rightarrow \Delta, \psi}{\Phi, \neg\psi \Rightarrow \Delta} \qquad (\Rightarrow \neg)\ \frac{\Phi, \psi \Rightarrow \Delta}{\Phi \Rightarrow \Delta, \neg\psi}$$

$$(\wedge \Rightarrow)\ \frac{\Phi, \psi, \varphi \Rightarrow \Delta}{\Phi, \psi \wedge \varphi \Rightarrow \Delta} \qquad (\Rightarrow \wedge)\ \frac{\Phi \Rightarrow \Delta, \psi \qquad \Phi \Rightarrow \Delta, \varphi}{\Phi \Rightarrow \Delta, \psi \wedge \varphi}$$

$$(\vee \Rightarrow)\ \frac{\Phi, \varphi \Rightarrow \Delta \qquad \Phi, \psi \Rightarrow \Delta}{\Phi, \varphi \vee \psi \Rightarrow \Delta} \qquad (\Rightarrow \vee)\ \frac{\Phi \Rightarrow \Delta, \varphi, \psi}{\Phi \Rightarrow \Delta, \varphi \vee \psi}$$

$$(\rightarrow \Rightarrow)\ \frac{\Phi \Rightarrow \Delta, \varphi \qquad \Phi, \psi \Rightarrow \Delta}{\Phi, \varphi \rightarrow \psi \Rightarrow \Delta} \qquad (\Rightarrow \rightarrow)\ \frac{\Phi, \varphi \Rightarrow \Delta, \psi}{\Phi \Rightarrow \Delta, \varphi \rightarrow \psi}$$

Beispiel

Das Ziel ist der Beweis folgender Aussage:

$(X \wedge Y) \rightarrow (X \vee Y)$ ist eine Tautologie.

Wir beweisen, dass die Sequenz $\Rightarrow (X \wedge Y) \rightarrow (X \vee Y)$ gültig ist.

Beweis.

$$\dfrac{\dfrac{\dfrac{\dfrac{}{X, Y \Rightarrow X, Y}}{X, Y \Rightarrow X \vee Y}}{X \wedge Y \Rightarrow X \vee Y}}{\Rightarrow (X \wedge Y) \rightarrow (X \vee Y)}$$

Beispiel

Wir beweisen die Gültigkeit folgender Sequenz:

$$\Rightarrow (X \to Y) \to (\neg Y \to \neg X)$$

Beweis.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{X, \neg Y \Rightarrow X}}{\neg Y \Rightarrow X, \neg X}}{\Rightarrow X, (\neg Y \to \neg X)} \qquad \dfrac{\dfrac{\dfrac{\dfrac{}{Y \Rightarrow Y, \neg X}}{Y, \neg Y \Rightarrow \neg X}}{Y \Rightarrow (\neg Y \to \neg X)}}{(X \to Y) \Rightarrow (\neg Y \to \neg X)}}{\Rightarrow (X \to Y) \to (\neg Y \to \neg X)}$$

Vollständigkeit und Korrektheit

Der Sequenzenkalkül ist vollständig und korrekt.

Korrektheit.

Nur gültige Sequenzen können im Sequenzenkalkül hergeleitet werden.

Vollständigkeit.

Alle gültigen Sequenzen können im Sequenzenkalkül hergeleitet werden.

Korrektheit des Sequenzenkalküls

Definition 4.54. (Korrektheit)

1. Eine Regel
$$\frac{\Phi \Rightarrow \Delta}{\Phi' \Rightarrow \Delta'}$$
ist korrekt, wenn aus der Gültigkeit von $\Phi \Rightarrow \Delta$ auch immer die Gültigkeit von $\Phi' \Rightarrow \Delta'$ folgt.

2. Eine Regel
$$\frac{\Phi_1 \Rightarrow \Delta_1 \qquad \Phi_2 \Rightarrow \Delta_2}{\Phi \Rightarrow \Delta}$$
ist korrekt, wenn aus der Gültigkeit von $\Phi_1 \Rightarrow \Delta_1$ und $\Phi_2 \Rightarrow \Delta_2$ die Gültigkeit von $\Phi \Rightarrow \Delta$ folgt.

Korrektheit des Sequenzenkalküls

Lemma 4.55. Die Regeln des Sequenzenkalküls sind korrekt.

Wir werden eine etwas stärkere Behauptung beweisen.

Definition 4.56. Sei $\Phi \Rightarrow \Delta$ eine Sequenz und β eine zu $\Phi \cup \Delta$ passende Belegung.

1. β falsifiziert die Sequenz $\Phi \Rightarrow \Delta$, falls sie alle $\varphi \in \Phi$ erfüllt aber kein $\delta \in \Delta$.
2. β erfüllt die Sequenz $\Phi \Rightarrow \Delta$, falls sie ein $\varphi \in \Phi$ nicht erfüllt oder aber mindestens ein $\delta \in \Delta$ erfüllt.

Lemma 4.57. Für jede Regel des Sequenzenkalküls und jede Belegung β die zu allen Formeln der Regel passt gilt:

β erfüllt die Konsequenz einer Regel genau dann, wenn β alle Prämissen erfüllt.

Beweis des Lemmas

Fall 1: (Disjunktion)

$$(\Rightarrow \vee)\ \frac{\Phi \Rightarrow \Delta, \varphi, \psi}{\Phi \Rightarrow \Delta, \varphi \vee \psi}$$

Beweis. Sei β eine zu allen Formeln der Regel passende Belegung.

Wir zeigen, dass β die Sequenz $\Phi \Rightarrow \Delta, \varphi, \psi$ genau dann falsifiziert, wenn β die Sequenz $\Phi \Rightarrow \Delta, \varphi \vee \psi$ falsifiziert.

$\Rightarrow$ Angenommen, β falsifiziert die Sequenz $\Phi \Rightarrow \Delta, \varphi, \psi$.

Also gilt $\beta \models \varphi$ für alle $\varphi \in \Phi$ aber $\beta \not\models \delta$ für alle $\delta \in \Delta, \varphi, \psi$.

Dann gilt aber $\beta \not\models \varphi \vee \psi$ und somit falsifiziert β die Sequenz $\Phi \Rightarrow \Delta, \varphi \vee \psi$.

$\Leftarrow$ Angenommen, β falsifiziert die Sequenz $\Phi \Rightarrow \Delta, \varphi \vee \psi$.

Also gilt $\beta \models \varphi$ für alle $\varphi \in \Phi$ aber $\beta \not\models \delta$ für alle $\delta \in \Delta, \varphi \vee \psi$.

Dann gilt aber $\beta \not\models \varphi$ und $\beta \not\models \psi$ und somit falsifiziert β die Sequenz $\Phi \Rightarrow \Delta, \varphi, \psi$.

Beweis des Lemmas

Fall 2. Disjunktion links.

$$(\vee \Rightarrow)\ \frac{\Phi, \varphi \Rightarrow \Delta \qquad \Phi, \psi \Rightarrow \Delta}{\Phi, \varphi \vee \psi \Rightarrow \Delta}$$

Sei β eine Belegung, die zu allen Formeln der Regel passt.

Wir zeigen, dass β die Sequenz $\Phi, \varphi \vee \psi \Rightarrow \Delta$ genau dann falsifiziert, wenn β $\Phi, \varphi \Rightarrow \Delta$ oder $\Phi, \psi \Rightarrow \Delta$ falsifiziert.

$\Rightarrow$ Wenn β die Sequenz $\Phi, \varphi \vee \psi \Rightarrow \Delta$ falsifiziert, gilt also $\beta \models \varphi'$ für alle $\varphi' \in \Phi \cup \{\varphi \vee \psi\}$ und $\beta \not\models \delta$ für alle $\delta \in \Delta$.

Es gilt also $\beta \models \varphi$ oder $\beta \models \psi$. Im ersten Fall wird aber die erste Prämisse falsifiziert, im zweiten Fall die zweite Prämisse.

$\Leftarrow$ Wenn β die Sequenzen $\Phi, \varphi \Rightarrow \Delta$ und $\Phi, \psi \Rightarrow \Delta$ falsifiziert, gilt also $\beta \models \varphi'$ für alle $\varphi' \in \Phi \cup \{\varphi, \psi\}$ und $\beta \not\models \delta$ für alle $\delta \in \Delta$.

Es gilt also auch $\beta \models \varphi \vee \psi$ und somit wird die Konlusion falsifiziert.

Korrektheit des Sequenzenkalküls

Lemma 4.58. Für jede Regel des Sequenzenkalküls und jede Belegung β die zu allen Formeln der Regel passt gilt:

β erfüllt die Konsequenz einer Regel genau dann, wenn β alle Prämissen erfüllt.

Folgerung 4.59. Die Regeln des Sequenzenkalküls sind korrekt.

Theorem 4.60. Jede im Sequenzenkalkül beweisbare Sequenz ist gültig.

Sequenzenkalkül

Ein Vorteil des Sequenzenkalküls ist, dass er eine systematische Beweissuche erlaubt.

Wir werden einen Algorithmus angeben, der zu jeder Sequenz $\Phi \Rightarrow \Delta$ entweder,

- einen Beweis im Sequenzenkalkül konstruiert oder
- eine Belegung, die alle Konsequenzen falsifiziert aber alle Voraussetzungen erfüllt.

Im zweiten Fall ist die Belegung ein Beweis, dass die Sequenz nicht gültig ist.

Beispiel

Erinnern wir uns an den Sequenzenkalkülbeweis der folgenden Aussage.

$$\Rightarrow (X \to Y) \to (\neg Y \to \neg X)$$

Beweis.

$$\dfrac{\dfrac{\dfrac{\dfrac{}{X, \neg Y \Rightarrow X}}{\neg Y \Rightarrow X, \neg X}}{\Rightarrow X, (\neg Y \to \neg X)} \qquad \dfrac{\dfrac{\dfrac{}{Y \Rightarrow Y, \neg X}}{Y, \neg Y \Rightarrow \neg X}}{Y \Rightarrow (\neg Y \to \neg X)}}{\dfrac{(X \to Y) \Rightarrow (\neg Y \to \neg X)}{\Rightarrow (X \to Y) \to (\neg Y \to \neg X)}}$$

Beispiel

Das folgende ist ein anderer Beweis der gleichen Aussage.

$$\Rightarrow (X \to Y) \to (\neg Y \to \neg X)$$

Beweis.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{X \Rightarrow X, Y} \quad \dfrac{}{X, Y \Rightarrow Y}}{(X \to Y), X \Rightarrow Y}}{(X \to Y), \Rightarrow \neg X, Y}}{(X \to Y), \neg Y \Rightarrow \neg X}}{(X \to Y) \Rightarrow (\neg Y \to \neg X)}}{\Rightarrow (X \to Y) \to (\neg Y \to \neg X)}$$

- Es gibt mehrere Beweis der gleichen Sequenz.
- Aber, für jede nicht-atomare Formel in einer Sequenz gibt es genau eine anwendbare Regel.

Beispiel

Wir wollen die folgende Sequenz beweisen $X \vee Y \Rightarrow X \wedge Y$.

Beweisversuch.

$$\dfrac{\dfrac{\overline{X \Rightarrow X} \quad Y \Rightarrow X}{X \vee Y \Rightarrow X} \qquad \dfrac{X \Rightarrow Y \quad \overline{Y \Rightarrow Y}}{X \vee Y \Rightarrow Y}}{X \vee Y \Rightarrow X \wedge Y}$$

- Der Beweisversuch liefert einen Baum dessen Blätter all mit Sequenzen beschriftet sind, die nur noch atomare Formeln enthalten. Es sind also keine weiteren Regeln mehr anwendbar.
- Aber nur zwei Blätter sind Axiome. Die anderen können falsifiziert werden.
- Wählen wir β durch $\beta(X) := 1$ und $\beta(Y) := 0$.
- Dies falsifiziert ein Blatt aber auch die Originalsequenz.
- D.h., der Versuch die Sequenz $X \vee Y \Rightarrow X \wedge Y$ zu beweisen führt zu einer falsifizierenden Belegung eines Blattes.
- Nach eben bewiesenem Lemma, falsifiziert dies auch die Originalsequenz.

Vollständigkeit des Sequenzenkalküls

Definition 4.61. Ein Ableitungsbaum für eine Sequenz $\mathcal{S}$ ist ein Baum, dessen Knoten wie folgt mit Sequenzen beschriftet sind:

- Jeder innere Knoten ist mit der Konsequenz einer Regel beschriftet. Die Kinder des Knotens sind mit den Prämissen der Regel beschriftet.
- Die Wurzel des Baumes ist mit $\mathcal{S}$ beschriftet.

Ein Blatt des Baums heißt positiv, wenn es mit einem Axiom beschriftet ist.

Es ist negativ, wenn es mit einer Sequenz $\Phi \Rightarrow \Delta$ beschriftet ist, so dass Φ, Δ disjunkte Mengen von Variablen sind.

Ein Ableitungsbaum für $\mathcal{S}$ ist vollständig, wenn alle Blätter positiv oder negativ sind.

Ein vollständiger Ableitungsbaum für $\mathcal{S}$ ist eine Widerlegung, wenn er ein negatives Blatt enthält.

Bemerkung. Ein Beweis für $\mathcal{S}$ ist ein vollständiger Ableitungsbaum dessen Blätter alle positiv sind.

Algorithmus zur Konstruktion von Beweisen

Algorithm 4.62.

Eingabe: Eine Sequenz $\Phi \Rightarrow \Delta$

Ausgabe: Ein Beweis für $\Phi \Rightarrow \Delta$ oder eine falsifizierende Belegung der Sequenz.

Wir bauen induktiv einen Ableitungsbaum für $\Phi \Rightarrow \Delta$ wie folgt:

Zunächst besteht der Baum aus einem mit $\Phi \Rightarrow \Delta$ beschrifteten Knoten.

Solange es unmakrierte Blätter gibt:

1. Wähle unmarkiertes Blatt t. Sei $\Phi' \Rightarrow \Delta'$ die Beschriftung.
2. Ist t negativ, konstruiere Belegung die alle Variablen in Φ' mit 1 belegt und alle anderen mit 0. Stop.
3. Ist t positiv, markiere es mit $(+)$.
4. Anderenfalls wähle nicht-atomare Formel $\varphi \in \Phi' \cup \Delta'$ und wende die (eindeutig bestimmte) anwendbare Regel an.
5. Füge ein oder zwei Nachfolger zu t hinzu, beschriftet mit den Prämissen der Regel mit Konklusion $\Phi' \Rightarrow \Delta'$

Sind alle Blätter markiert, gib dem Baum als Beweis für $\Phi \Rightarrow \Delta$ zurück.

Vollständigkeit des Sequenzenkalküls

Theorem 4.63. Der Algorithmus terminiert auf jeder Sequenz $\Phi \Rightarrow \Delta$, vorausgesetzt das $\Phi \cup \Delta$ endlich ist.

Er findet einen Beweis für $\Phi \Rightarrow \Delta$, so es einen gibt.

Anderenfalls gibt er eine Widerlegung für $\Phi \Rightarrow \Delta$ zurück.

Beweis. Wir zeigen zunächst Terminierung.

- Sei die Komplexität einer Sequenz $\Phi \Rightarrow \Delta$ definiert als die Zahl der Verknüpfungen, die in Formeln aus $\Phi \cup \Delta$ vorkommen.
- Jede Regel des Sequenzenkalküls erhöht die Komplexität, d.h. die Konklusion einer Regel ist komplexer als ihre Prämissen.
- Der Algorithmus muss also terminieren.

Vollständigkeit des Sequenzenkalküls

Theorem 4.64. Der Algorithmus terminiert auf jeder Sequenz $\Phi \Rightarrow \Delta$, vorausgesetzt das $\Phi \cup \Delta$ endlich ist.

Er findet einen Beweis für $\Phi \Rightarrow \Delta$, so es einen gibt.

Anderenfalls gibt er eine Widerlegung für $\Phi \Rightarrow \Delta$ zurück.

Beweis (Teil 2). Angenommen der Algorithmus findet einen Ableitungsbaum, in dem alle Blätter markiert und somit positiv sind.

- Dann ist der Baum ein Beweis im Sequenzenkalkül und nach dem Korrektheitssatz ist die Sequenze $\Phi \Rightarrow \Delta$ gültig.

Anderenfalls findet der Algorithmus ein negatives Blatt und konstruiert eine falsifizierende Belegung.

- Nach dem Lemma, falsifiziert diese Belegung auch $\Phi \Rightarrow \Delta$.

□

Vollständigkeit des Sequenzenkalküls

Definition 4.65. Sei $\Phi \subseteq \mathrm{AL}$ eine Menge von Formeln.

Eine Formel $\psi \in \mathrm{AL}$ kann aus Φ hergeleitet werden, geschrieben $\Phi \vdash_S \psi$, wenn es eine endliche Teilmenge $\Phi' \subseteq \Phi$ gibt, so dass $\Phi' \Rightarrow \psi$ im Sequenzenkalkül beweisbar ist.

Insbesondere kann ψ aus der leeren Menge hergeleitet werden, d.h. $\vdash_S \psi$, wenn $\emptyset \Rightarrow \psi$ im Sequenzenkalkül beweisbar ist.

Definition 4.66. Sei $\Phi \subseteq \mathrm{AL}$ eine endliche oder unendliche Formelmenge. Φ ist inkonsistent, wenn es eine Formel $\psi \in \mathrm{AL}$ gibt, so dass $\Phi \vdash_S \psi$ und $\Phi \vdash_S \neg\psi$.

Anderenfalls ist Φ konsistent.

Vollständigkeit des Sequenzenkalküls

Das vorherige Theorem und der Algorithmus zeigen die Vollständigkeit des Sequenzenkalküls für endliche Formelmengen.

Mit Hilfe des Kompaktheitssatzes folgt die Vollständigkeit für beliebige Formelmengen.

Theorem 4.67. (Vollständigkeit und Korrektheit)

Sei $\Phi \subseteq \mathrm{AL}$ eine Menge von Formeln und sei $\psi \in \mathrm{AL}$.

1. Φ ist konsistent genau dann, wenn Φ erfüllbar ist.
2. $\Phi \vdash_S \psi$ genau dann, wenn $\Phi \models \psi$.

Ableitbarkeit vs. Semantische Folgerung

Sequenzenkalkül	Semantische Konzepte
Sequenz $\Phi \Rightarrow \psi$ ist gültig	Aus Φ folgt logisch ψ
$\vdash$	$\models$
Φ ist konsistent	Φ ist erfüllbar
$\emptyset \Rightarrow \psi$ gültig	ψ allgmeingültig
$\psi \Rightarrow \emptyset$ gültig	ψ unerfüllbar

Erweiterung des Sequenzenkalküls auf FO

Wir erweitern den Sequenzenkalkül für die Prädikatenlogik.

Dazu können wir alle bisherigen Regeln und Definitionen direkt übernehmen.

Wir brauchen nur noch Regeln für die Quantoren.

Dies erfordert allerdings etwas Vorarbeit.

Elimination von Variablen

Bei der Definition von Substitutionen haben wir gesehen, dass die Kombination von freien und gebundenen Variablen Probleme bereiten kann.

Um ähnliche Probleme im Sequenzenkalkül zu vermeiden, werden wir freie Variablen durch Konstantensymbole substituieren und somit nur mit Sätzen arbeiten.

Lemma. Sei σ eine Signatur.

Sei $\varphi(x_1, \ldots, x_k)$ eine Formel mit freien Variablen $\mathit{frei}(\varphi) \subseteq \{x_1, \ldots, x_k\}$.

Seien $c_1, \ldots, c_k$ Konstantensymbole mit $c_i \notin \sigma$ für alle $1 \leqslant i \leqslant k$.

φ ist erfüllbar gdw. $\varphi[x_1/c_1, \ldots, x_k/c_k]$ erfüllbar ist.

Beweis. Wenn φ erfüllbar ist, dann gibt es eine σ-Struktur $\mathcal{A}$ und Belegung β mit $\{x_1, \ldots, x_k\} \subseteq \mathrm{Dom}(\beta)$, so dass $(\mathcal{A}, \beta) \models \varphi$.

Aber dann erfüllt die $\sigma \cup \{c_1, \ldots, c_k\}$-Struktur $\mathcal{B}$ mit $\mathcal{B}_{|\sigma} = \mathcal{A}$ und $c_i^{\mathcal{B}} := \beta(x_i)$ die Formel $\varphi[x_1/c_1, \ldots, x_k/c_k]$. Die Umkehrung ist analog.

Elimination von Variablen

Das Lemma zeigt, dass Erfüllbarkeit, Gültigkeit, ... von Formeln auf entsprechende Aussagen über Sätzen reduziert werden können.

Notation. In der Formulierung des Sequenzenkalküls werden wir ausschließlich mit Sätzen arbeiten, d.h. Formeln ohne freie Variablen.

Das heißt, wann immer wir $\Phi, \Delta, \varphi, \psi, \ldots$ schreiben, meinen wir Sätze oder Mengen von Sätzen.

Ausname. Wenn wir $\varphi(x)$ oder $\psi(x)$ schreiben, dann soll das bedeuten, dass die Formeln eine freie Variable x haben.

Hinweis. Für den Rest des Abschnitts fixieren wir eine Signatur σ und eine anzählbar unendliche Menge $c_0, c_1, \ldots$ von Konstantensymbolen $c_i \notin \sigma$, für alle $i \geqslant 0$.

Das heißt, formal arbeiten wir in der Signatur $\tau := \sigma \cup \{c_0, c_1, \ldots\}$.

Sequenzen

Analog zum aussagenlogischen Sequenzenkalkül definieren wir Sequenzen, ... für die Prädikatenlogik.

Definition 4.68. Sei σ eine Signatur.

1. Eine Sequenz ist eine Aussage der Form

$$\Phi \Rightarrow \Delta$$

   wobei $\Phi, \Delta \subseteq \mathrm{FO}[\sigma]$ Mengen von Formeln sind. Wir nennen Φ die Voraussetzungen und Δ die Konklusionen.
2. Eine Sequenz $\Phi \Rightarrow \Delta$ ist gültig, wenn jede σ-Interpretation $\mathcal{I}$, die alle Formeln in Φ erfüllt auch eine Formel aus Δ erfüllt.
3. Wenn $\Phi \Rightarrow \Delta$ nicht gültig ist, dann gibt es eine σ-Interpretation $\mathcal{I}$, die alle $\varphi \in \Phi$ aber kein $\psi \in \Delta$ erfüllt.

   Wir sagen: $\mathcal{I}$ falisifiziert die Sequenz.

Axiome und Regeln

Definition 4.69. Ein Axiom des Sequenzenkalküls ist eine Sequenz $\Phi \Rightarrow \Delta$, so dass $\Phi \cap \Delta \neq \emptyset$.

Die Regeln des Sequenzenkalküls.
Die Regeln des Sequenzenkalküls sind die Regeln des aussagenlogischen Kalküls erweitert um die Gleichheitsregel, die Substitutionsregel und die Quantorenregeln.

Die Regeln für Gleichheit und Substitution

Die Regel für Gleichheit.

$$(=)\ \frac{\Phi, t = t \Rightarrow \Delta}{\Phi \Rightarrow \Delta}$$

Die Regeln der Substitution.

$$(S \Rightarrow)\ \frac{\Phi, \psi(t) \Rightarrow \Delta}{\Phi, t \doteq t', \psi(t') \Rightarrow \Delta} \qquad (\Rightarrow S)\ \frac{\Phi, \Rightarrow \Delta, \psi(t)}{\Phi, t \doteq t' \Rightarrow \Delta, \psi(t')}$$

Die Notation $t \doteq t'$ bedeutet, dass wir $t = t'$ oder $t' = t$ verwenden können.

Hierbei sind t, t' Terme ohne freie Variablen und $\psi(x)$ ist eine Formel, wobei x die einzige freie Variable ist.

Beispiel. Sei $\sigma := \{R, f, c\}$, wobei R ein Relationssymbol, f ein Funktionssymbol und c ein Konstantensymbol ist.

$$\frac{Rfc \Rightarrow Rfc}{Rfc, fc = c \Rightarrow Rffc}$$

(setze $\psi(x) := Rfx$ und wende $(\Rightarrow S)$ an)

Die Quantorenregeln

Die Regeln für den Existenzquantor.

$$(\exists \Rightarrow)\ \frac{\Phi, \psi(c) \Rightarrow \Delta}{\Phi, \exists x \psi(x) \Rightarrow \Delta}$$

wenn c nicht in Φ, Δ und ψ vorkommt

$$(\Rightarrow \exists)\ \frac{\Phi \Rightarrow \Delta, \psi(t)}{\Phi \Rightarrow \Delta, \exists x \psi(x)}$$

Beispiel. Das folgende ist ein Beweis für $\exists x \exists y Rxy \Rightarrow \exists y \exists x Rxy$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{R(c,d) \Rightarrow R(c,d)}}{R(c,d) \Rightarrow \exists x R(x,d)}}{R(c,d) \Rightarrow \forall y \exists x R(x,y)}}{\exists y R(c,y) \Rightarrow \exists y \exists x R(x,y)}}{\exists x \exists y R(x,y) \Rightarrow \exists y \exists x R(x,y)}$$

Die Quantorenregeln

Die Regeln für den Allquantor.

$$(\forall \Rightarrow) \ \frac{\Phi, \psi(t) \Rightarrow \Delta}{\Phi, \forall x \psi(x) \Rightarrow \Delta}$$

$$(\Rightarrow \forall) \ \frac{\Phi \Rightarrow \Delta, \psi(c)}{\Phi \Rightarrow \Delta, \forall x \psi(x)}$$

wenn c nicht in Φ, Δ und $\psi(x)$ vorkommt

Beispiel. Das folgende ist ein Beweis für $\forall x \exists y Rxy \Rightarrow \exists y \forall x Rxy$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\overline{R(c,d) \Rightarrow R(c,d)}}{R(c,d) \Rightarrow \exists x R(x,d)}}{\forall y R(c,y) \Rightarrow \exists x R(x,d)}}{\forall y R(c,y) \Rightarrow \forall y \exists x R(x,y)}}{\exists x \forall y R(x,y) \Rightarrow \forall y \exists x R(x,y)}$$

Die Regeln des prädikatenlogischen Sequenzenkalküls

$$(\neg \Rightarrow)\ \frac{\Phi \Rightarrow \Delta, \psi}{\Phi, \neg\psi \Rightarrow \Delta} \qquad (\Rightarrow \neg)\ \frac{\Phi, \psi \Rightarrow \Delta}{\Phi \Rightarrow \Delta, \neg\psi}$$

$$(\wedge \Rightarrow)\ \frac{\Phi, \psi, \varphi \Rightarrow \Delta}{\Phi, \psi \wedge \varphi \Rightarrow \Delta} \qquad (\Rightarrow \wedge)\ \frac{\Phi \Rightarrow \Delta, \psi \qquad \Phi \Rightarrow \Delta, \varphi}{\Phi \Rightarrow \Delta, \psi \wedge \varphi}$$

$$(\vee \Rightarrow)\ \frac{\Phi, \varphi \Rightarrow \Delta \qquad \Phi, \psi \Rightarrow \Delta}{\Phi, \varphi \vee \psi \Rightarrow \Delta} \qquad (\Rightarrow \vee)\ \frac{\Phi \Rightarrow \Delta, \varphi, \psi}{\Phi \Rightarrow \Delta, \varphi \vee \psi}$$

$$(\rightarrow\Rightarrow)\ \frac{\Phi \Rightarrow \Delta, \varphi \qquad \Phi, \psi \Rightarrow \Delta}{\Phi, \varphi \rightarrow \psi \Rightarrow \Delta} \qquad (\Rightarrow\rightarrow)\ \frac{\Phi, \varphi \Rightarrow \Delta, \psi}{\Phi \Rightarrow \Delta, \varphi \rightarrow \psi}$$

$$(\forall \Rightarrow)\ \frac{\Phi, \psi(t) \Rightarrow \Delta}{\Phi, \forall x \psi(x) \Rightarrow \Delta} \qquad (\Rightarrow \forall)\ \frac{\Phi \Rightarrow \Delta, \psi(c)}{\Phi \Rightarrow \Delta, \forall x \psi(x)}\ \textit{c n. i.}\ \Phi, \Delta, \psi$$

$$(\exists \Rightarrow)\ \frac{\Phi, \psi(c) \Rightarrow \Delta}{\Phi, \exists x \psi(x) \Rightarrow \Delta}\ \textit{c n.i.}\ \Phi, \Delta, \psi \qquad (\Rightarrow \exists)\ \frac{\Phi \Rightarrow \Delta, \psi(t)}{\Phi \Rightarrow \Delta, \exists x \psi(x)}$$

$$(S \Rightarrow)\ \frac{\Phi, \psi(t) \Rightarrow \Delta}{\Phi, t \doteq t', \psi(t') \Rightarrow \Delta} \qquad (\Rightarrow S)\ \frac{\Phi, \Rightarrow \Delta, \psi(t)}{\Phi, t \doteq t' \Rightarrow \Delta, \psi(t')} \qquad (=)\ \frac{\Phi, t = t \Rightarrow \Delta}{\Phi \Rightarrow \Delta}$$

Beispiel

Beispiel. Wir zeigen $\varphi \vee \exists x \varphi(x) \equiv \exists x(\varphi \vee \psi(x))$, falls x nicht in φ vorkommt.

Sequenzenkalkülbeweis von $\varphi \vee \exists x \varphi(x) \Rightarrow \exists x(\varphi \vee \psi(x))$.

$$\dfrac{\dfrac{\dfrac{}{\varphi \Rightarrow \varphi, \psi(c)}}{\varphi \Rightarrow \varphi \vee \psi(c)}}{\varphi \Rightarrow \exists x(\varphi \vee \psi(x))} \qquad \dfrac{\dfrac{\dfrac{\dfrac{}{\psi(c) \Rightarrow \varphi, \psi(c)}}{\psi(c) \Rightarrow \varphi \vee \psi(c)}}{\psi(c) \Rightarrow \exists x(\varphi \vee \psi(x))}}{\exists x \psi(x) \Rightarrow \exists x(\varphi \vee \psi(x))}$$

$$\dfrac{\varphi \Rightarrow \varphi, \psi(c) \qquad \psi(c) \Rightarrow \varphi, \psi(c)}{\varphi \vee \exists x \psi(x) \Rightarrow \exists x(\varphi \vee \psi(x))}$$

Der Sequenzenkalkülbeweis von $\exists x(\varphi \vee \psi(x)) \Rightarrow \varphi \vee \exists x \varphi(x)$ ist ähnlich.

Die Annahme, dass x nicht in φ vorkommt, wird im linken Zweig benutzt, da wir sonst folgenden nicht-Beweis erhalten

$$\dfrac{\dfrac{\varphi(x) \Rightarrow \varphi(c), \psi(c)}{\varphi(x) \Rightarrow \varphi(c) \vee \psi(c)}}{\varphi(x) \Rightarrow \exists x(\varphi(x) \vee \psi(x))}$$

Korrektheit der Regeln

Wir definieren die Korrektheit der prädikatenlogischen Regeln genauso wie für die Aussagenlogik.

Definition 4.70. Eine Regel

$$\frac{\Phi \Rightarrow \Delta}{\Phi' \Rightarrow \Delta'}$$

ist korrekt, wenn aus der Gültigkeit von $\Phi \Rightarrow \Delta$ die Gültigkeit von $\Phi' \Rightarrow \Delta'$ folgt.

- Um die Korrektheit einer Regel nachzuweisen, müssen wir zeigen, dass jede Interpretation, die alle Formeln in Φ' erfüllt auch eine Formel aus Δ erfüllt.
- Für den Beweis können wir benutzen, dass $\Phi \Rightarrow \Delta$ gültig ist.

Korrektheit der Quantorenregeln

Die Regeln für den Allquantor.

$$(\forall \Rightarrow) \frac{\Phi, \psi(t) \Rightarrow \Delta}{\Phi, \forall x \psi(x) \Rightarrow \Delta}$$

$$(\Rightarrow \forall) \frac{\Phi \Rightarrow \Delta, \psi(c)}{\Phi \Rightarrow \Delta, \forall x \psi(x)}$$

wenn c nicht in Γ, Δ and $\psi(x)$ vorkommt

Lemma 4.71. Die Regeln für den Allquantor sind korrekt.

Korrektheit der $(\forall \Rightarrow)$ Regel

$$(\forall \Rightarrow) \frac{\Phi, \psi(t) \Rightarrow \Delta}{\Phi, \forall x \psi(x) \Rightarrow \Delta}$$

Behauptung. Die $(\forall \Rightarrow)$ Regel ist korrekt.

Sei $\mathfrak{I} := (\mathcal{A}, \beta)$ eine σ-Interpretation, so dass $\mathfrak{I} \models \Phi$ und $\mathfrak{I} \models \forall x \psi(x)$.

Sei $a := [\![t]\!]^{\mathfrak{I}}$. Dann gilt also $\mathfrak{I} \models \psi[a]$.

Es folgt, dass $\mathfrak{I} \models \psi(t)$ und, da nach Voraussetzung $\Phi, \psi(t) \Rightarrow \Delta$ gültig ist, $\mathfrak{I} \models \delta$ für ein $\delta \in \Delta$. $\square$

Korrektheit der $(\Rightarrow \forall)$ Regel

$$(\Rightarrow \forall)\ \frac{\Phi \Rightarrow \Delta, \psi(c)}{\Phi \Rightarrow \Delta, \forall x \psi(x)}$$

wenn c nicht in Γ, Δ und $\psi(x)$ vorkommt

Behauptung. Die $(\Rightarrow \forall)$ Regel ist korrekt.

Sei τ die Signatur, die alle (Funktions-, Relations- und Konstanten-) Symbole enthält, die in $\Phi, \Delta, \psi(x)$ vorkommen, aber nicht c.

Sei $\mathcal{I} := (\mathcal{A}, \beta)$ eine τ-Interpretation mit $\mathcal{I} \models \Phi$.

1. Wenn $\mathcal{I} \models \delta$, für ein $\delta \in \Delta$, dann sind wir fertig.
2. Anderenfalls, $\mathcal{I} \not\models \delta$ für alle $\delta \in \Delta$.

   Für $a \in A$ sei $\mathcal{I}_a$ die $\tau \cup \{c\}$-Expansion von $\mathcal{I}$ mit $c^{\mathcal{I}_a} := a$ (d.h. $(\mathcal{I}_a)_{|\tau} = \mathcal{I}$).

   Da c nur in ψ vorkommt, folgt aus dem Koinzidenzlemma, dass $\mathcal{I}_a \models \Phi$ aber $\mathcal{I}_a \not\models \delta$ für alle $\delta \in \Delta$. Also gilt $\mathcal{I}_a \models \psi(c)$.

   Es folgt, dass $\mathcal{I}_a \models \psi(c)$ für alle $a \in A$ und somit $\mathcal{I} \models \forall x \psi(x)$. □

Korrektheit der $(\forall \Rightarrow)$ Regel

$$(\Rightarrow \forall)\ \frac{\Phi \Rightarrow \Delta, \psi(c)}{\Phi \Rightarrow \Delta, \forall x \psi(x)}$$ wenn c nicht in Γ, Δ und $\psi(x)$ vorkommt

Die Voraussetzung "c kommt nicht in Γ, Δ und $\psi(x)$ vor" ist notwendig.

Beispiel.

Sei $\Delta := \emptyset$ und $\Phi := R(c)$ und $\psi(x) := R(x)$.

Dann ist $\Phi \Rightarrow \psi(c)$ sicherlich gültig.

Aber $\Phi \Rightarrow \forall x R(x)$ ist nicht gültig, wie die folgende Struktur $\mathcal{A} := (\{0, 1\}, c^{\mathcal{A}} := 0, R^{\mathcal{A}} := \{0\})$ zeigt.

Beweise im Sequenzenkalkül

Definition 4.72. Ein Beweis im Sequenzenkalkül ist ein Baum, dessen Knoten wie folgt mit Sequenzen beschriftet sind:

- Die Blätter sind mit Axiomen beschriftet.
- Jeder innere Knoten ist mit der Konsequenz einer Regel beschriftet und dessen Kinder mit deren Prämissen. Jeder innere Knoten hat also entweder ein oder zwei Kinder.

Definition 4.73.

1. Eine Sequenz $\Phi \Rightarrow \Delta$ ist im Sequenzenkalkül beweisbar, oder ableitbar, wenn es als Beschriftung eines Knotens in einem Beweis vorkommt.
2. Eine Formel ψ ist aus einer Menge Φ von Formeln ableitbar oder beweisbar, geschrieben $\Phi \vdash \psi$, wenn $\Phi \Rightarrow \psi$ beweisbar ist.
3. Eine Menge $\Phi \subseteq \mathsf{FO}[\sigma]$ von Formeln ist inkonsistent, wenn es eine Formel $\varphi \in \mathsf{FO}[\sigma]$ gibt, so dass $\Phi \vdash \varphi$ und $\Phi \vdash \neg\varphi$.
4. Ansonsten ist Φ konsistent.

Beweise im Sequenzenkalkül

Bemerkung. Die Klasse der ableitbaren Sequenzen ist also die kleinste Klasse die alle Axiome enthält und wann immer sie die Prämissen einer Regel enthält, dann auch deren Konsequenz.

Korrektheit des Sequenzenkalküls

Theorem 4.74. (Korrektheitssatz)

Die Regeln des Sequenzenkalküls sind korrekt, d.h. jede beweisbare Sequenz ist gültig..

Beweis. Der Satz folgt sofort aus der Korrektheit der einzelnen Regeln.

Wir haben die Korrektheit der aussagenlogischen Regeln sowie der Regeln für den Allquantor bewiesen,

Die anderen Regeln sind zur Übung empfohlen.

Vollständigkeit und Korrektheit

Theorem 4.75.

1. Sei $\Phi \subseteq \mathsf{FO}[\sigma]$ eine Menge von Formeln und $\psi \in \mathsf{FO}[\sigma]$.

   $\Phi \models \psi$ genau dann, wenn $\Phi \Rightarrow \psi$ beweisbar ist.

2. Sei $\Phi \subseteq \mathsf{FO}[\sigma]$ eine Menge von Formeln.

   Φ ist erfüllbar genau dann, wenn Φ konsistent ist.

Beweis. Die Richtung von "rechts nach links" ist im wesentlichen der Korrektheitssatz.

Die andere Richtung ist erheblich schwieriger und ist als Gödel's Vollständigkeitssatz bekannt.

(Nicht zu verwechseln mit Gödel's Unvollständigkeitssatz.)

Ableitbarkeit vs. Semantische Folgerung

Sequenzenkalkül	Semantische Konzepte
Sequenz $\Phi \Rightarrow \psi$ ist gültig	Aus Φ folgt logisch ψ
$\vdash$	$\models$
Φ ist konsistent	Φ ist erfüllbar
$\emptyset \Rightarrow \psi$ gültig	ψ allgmeingültig
$\psi \Rightarrow \emptyset$ gültig	ψ unerfüllbar

Der Kompaktheitssatz

Theorem 4.76. (Kompaktheit der Prädikatenlogik)

1. Sei $\Phi \subseteq \mathsf{FO}[\sigma]$ eine Menge von Formeln.

   Φ ist erfüllbar genau dann, wenn jede endliche Teilmenge von Φ erfüllbar ist.

2. Sei $\Phi \subseteq \mathsf{FO}[\sigma]$ und $\psi \in \mathsf{FO}[\sigma]$.

   $\Phi \models \psi$ genau dann, wenn es eine endliche Teilmenge $\Phi_0 \subseteq \Phi$ gibt, so dass $\Phi_0 \models \psi$.

Beweis des Kompaktheitssatzes

Behauptung 1. Sei $\Phi \subseteq \mathsf{FO}[\sigma]$ eine Menge von Formeln.

Φ ist erfüllbar genau dann, wenn jede endliche Teilmenge von Φ erfüllbar ist.

Beweis. Wenn Φ erfüllbar ist, so natürlich auch jede endliche Teilmenge.

Sei also jede endliche Teilmenge von Φ erfüllbar. Angenommen, Φ sei nicht erfüllbar. Dann gibt es also ein $\psi \in \mathsf{FO}[\sigma]$, so dass $\Phi \models \psi$ und $\Phi \models \neg\psi$.

Aus dem Vollständigkeitssatz folgt nun, dass $\Phi \vdash \psi$ und $\Phi \vdash \neg\psi$. Beide beweise sind aber endlich und verwenden daher nur endlich viele Formeln aus Φ. Seien Φ_1, Φ_2 die in den beiden Beweisen benutzen Formeln aus Φ. Dann sind also ψ und $\neg\psi$ bereits aus $\Phi_1 \cup \Phi_2$ ableitbar und somit ist $\Phi_1 \cup \Phi_2$ inkonsistent und somit unerfüllbar, im Widerspruch zur Annahme. □

Der Kompaktheitssatz

Behauptung 2.

Sei $\Phi \subseteq \mathsf{FO}[\sigma]$ und $\psi \in \mathsf{FO}[\sigma]$.

$\Phi \models \psi$ genau dann, wenn es eine endliche Teilmenge $\Phi_0 \subseteq \Phi$ gibt, so dass $\Phi_0 \models \psi$.

Beweis. Die Behauptung folgt sofort aus dem ersten Teil. Denn $\Phi \models \psi$ gdw. $\Phi \cup \{\neg\psi\}$ unerfüllbar ist gdw. es eine unerfüllbare endliche Teilmenge $\Phi_0 \subseteq \Phi \cup \{\neg\psi\}$ gibt gdw. $\Phi_0 \setminus \{\neg\psi\} \models \psi$. □

Entscheidbarkeit der Aussagenlogik

Wir haben gesehen, dass es einen Algorithmus gibt, der als Eingabe eine aussagenlogische Sequenz $\Phi \Rightarrow \Delta$ bekommt und entscheidet, ob $\Phi \Rightarrow \Delta$ im aussagenlogischen Sequenzenkalkül beweisbar ist.

Der Algorithmus terminiert auf jeder endlichen Eingabe.

Insbesondere kann der Algorithmus benutzt werden, um für eine Formel $\psi \in \mathrm{AL}$ zu entscheiden, ob $\Rightarrow \psi$ beweisbar und somit ψ eine Tautologie ist

Theorem 4.77.

1. Das Allgemeingültigkeitsproblem der Aussagenlogik ist entscheidbar.
2. Das Erfüllbarkeitsproblem der Aussagenlogik ist entscheidbar.

Das klassische “Entscheidungsproblem”

Die Frage, on es einen solchen Algorithmus auch für die Prädikatenlogik gibt, war eines der wichtigsten Probleme der mathematischen Logik.

> *Das Entscheidungsproblem ist gelöst, wenn man ein Verfahren kennt, das bei einem vorgelegten logischen Ausdruck durch endlich viele Operationen die Entscheidung über die Allgemeingültigkeit bzw. Erfüllbarkeit erlaubt. (...) Das Entscheidungsproblem muss als das Hauptproblem der mathematischen Logik betrachtet werden.*

(D. Hilber, W. Ackermann: Grundzüge der theoretischen Logik, Vol. 1, Berlin 1928, p.73ff.

Das klassische Entscheidungsproblem

Eine positive Antwort auf das Entscheidungsproblem, also ein Algorithmus, der Allgemeingültigkeit oder Erfüllbarkeit prädikatenlogischer Formeln entscheiden könnte, hätte weitreichende Folgen für die Mathematik.

Wir könnten dann zum Beispiel die Gruppentheorie entscheiden, d.h. für jede Formel in der Signatur der Gruppen entscheiden, ob sie in allen Gruppen gilt.

Unentscheidbarkeit der Prädikatenlogik

Für die Prädikatenlogik kann ein solcher Algorithmus nicht existieren.

Theorem 4.78. (Church, Turing 1936, 1937)

Das Allgemeingültigkeits- und somit das Erfüllbarkeitsproblem der Prädikatenlogik ist unentscheidbar.

Das heißt, es gibt keinen Algorithmus, der auf Eingabe $\psi \in \mathsf{FO}$, korrekt entscheidet, ob ψ gültig ist.

Eine kurze Wiederholung des Teils über Logik

Zentrale Themen der Vorlesung bisher

Ziele der Vorlesung.

- Aussagen und Informationen präzise und eindeutig durch logische Formeln formalisieren zu können.
- Die Formalisierung in der Logik soll helfen, neue Schlüsse aus dem vorhandenen Wissen zu ziehen.
- Wir wollen Methoden haben, um solche Schlüsse in einem Beweissystem beweisen zu können.
- Methoden, um die Grenzen der Formalisierbarkeit in einer Logik bestimmen zu können.
- Frage nach der Komplexität und Entscheidbarkeit algorithmischer Probleme in der Logik.

Die Art der zu formalisierenden Aussagen bestimmt dabei die Logik.

- Aussagenlogik: Grundlagen logischen Schließens
- Prädikatenlogik: Aussagen über Strukturen
- Temporallogik: Aussagen über zeitliches Verhalten von Systemen

Zentrale Themen der Vorlesung

Zentrale Aspekte für jede Logik:

1. Syntax: Wie sieht die Logik/die Formeln aus.
2. Semantik: Was bedeuten die Formeln. Definition von $\mathcal{A} \models \varphi$
   - Semantik induktiv über den Formelaufbau
3. Formalisieren von Aussagen in der Logik
   - Axiomatisieren einer Strukturklasse
   - Modellklasse einer Menge von Formeln
   - Definieren einer Relation $\varphi(\mathcal{A})$

   ### Wie kann ich es in der Logik sagen.
   - Äquivalenzumformungen
   - Substitution, Ersetzungslemma
   - Normalformen (NNF, KNF, DNF, PNF)

   ### Was kann ich in der Logik überhaupt ausdrücken
   - Kompaktheitssatz
   - Ehrenfeucht-Fraïssé Spiele

Zentrale Themen der Vorlesung

Zentrale Aspekte einer Logik:

4. Erfüllbarkeit, Allgemeingültigkeit
5. Schließen in einer Logik: $\Phi \models \psi$

   Wie kann man Schlüsse in einem formalen Beweissystem beweisen
   - Sequenzenkalkül
   - Resolution
   - DPLL

   Reduktion von logischer Folgerung auf Erfüllbarkeit:

   $$\Phi \models \psi \text{ gdw. } \Phi \cup \{\neg\psi\} \text{ unerfüllbar.}$$

   Das liefert syntaktische Begriffe zu den entsprechenden semantisches Aussagen.
6. Algorithmische Probleme in der Logik
   - Entscheidbarkeit der Logik, d.h. des Erfüllbarkeitsproblems
   - Komplexität des Auswertens von Formeln

Griechische Symbole

Anbei eine Liste einiger häufig benutzter griechischer Symbole, teilweise in Klein- und Großschreibweise, sortiert nach Verwendungsarten.

α	alpha
β	beta
γ, Γ	gamma
δ, Δ	delta
ε	epsilon
φ, Φ	phi
ψ, Ψ	psi
θ, Θ	theta
χ	chi
ξ, Ξ	xi
λ, Λ	lambda
ω, Ω	omega
μ	mü
ν	nü