

Motivation
ooooo

Inhalte
oooo

Organisation
oooooo

logische Folgerungen
oooooooooooooooo

Zusammenfassung
oo

Logik für Informatiker:innen

Till Mossakowski

Sommersemester 2024

Überblick über die heutige Veranstaltung

- Warum ist Logik für Informatiker:innen wichtig?
- Inhalte der Veranstaltung
- Organisation der Veranstaltung
- Logik ist die Wissenschaft des Schlussfolgerns: “Was folgt aus welchem?”

Motivation
●○○○○

Inhalte
○○○○

Organisation
○○○○○

logische Folgerungen
○○○○○○○○○○○○○○○○

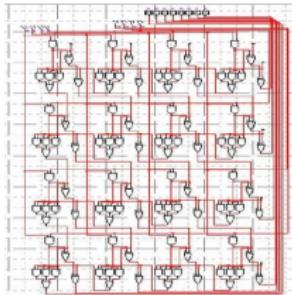
Zusammenfassung
○○

Warum ist Logik für eine:n Informatiker:in wichtig?

Schaltkreisentwurf und -verifikation



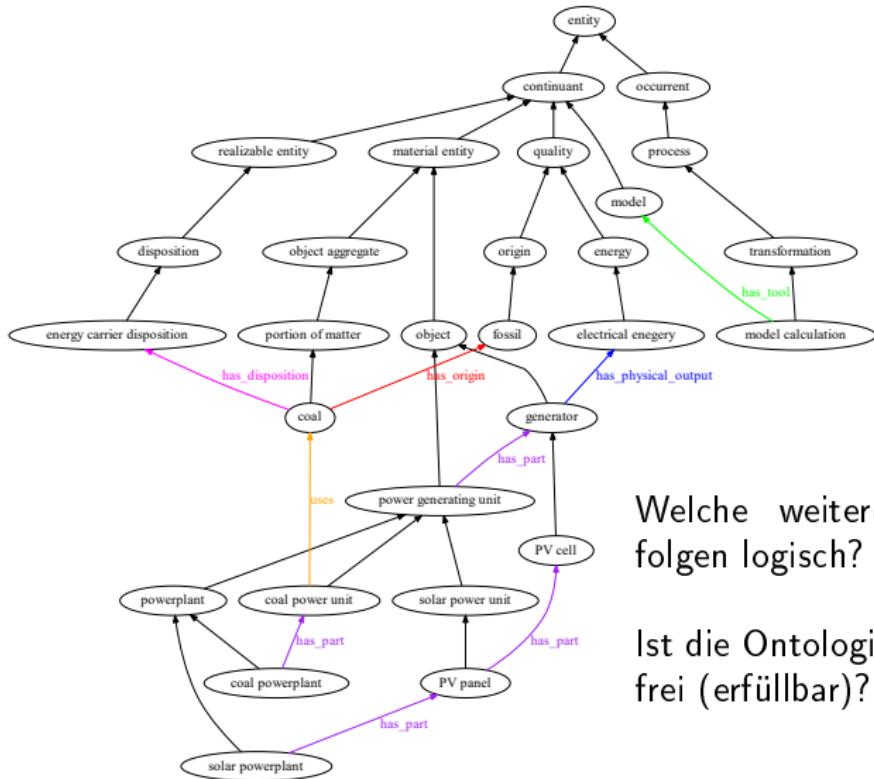
Bug in Pentium-Arithmetik
1994: ca. 500 Mio. \$ Kosten.
Seitdem wird Mikroprozessor-
arithmetik logisch verifiziert.



$$\models \forall x \forall y |(x \otimes y) - (x * y)| < \varepsilon$$

folgt logisch Korrektheitseigenschaft

Wissenrepräsentation mittels Ontologien



Welche weitere Beziehungen folgen logisch?

Ist die Ontologie widerspruchsfrei (erfüllbar)?

Logik im Studienplan

Semester	1	2	3	4	5	6
Prüfungen	8 CP	6 CP	5 CP			
Informatik I - Pflicht	Einführung in die Informatik (8 CP) mind. 5 CP benötigt Modellierung (5 CP) Datenbanken (5 CP)	Algorithmen und Datenstrukturen (6 CP)	Software Engineering + IT-PM (5 CP)			
Prüfungen		mind. 10 CP benötigt				
Informatik II - Pflicht		Programmierparadigmen (5 CP)	Intelligente Systeme (5 CP)	Sichere Systeme (5 CP)		
Prüfungen		mind. 20 CP benötigt				
Informatik - Wahlpflicht		WPF Informatik (5 CP)			WPF Informatik oder Mathematik (5 CP) WPF Informatik (5 CP) WPF Informatik (5 CP)	WPF Informatik (5 CP)
Prüfungen		mind. 5 CP benötigt				
Technische Informatik		Technische Informatik 1 (5 CP)	Technische Informatik 2 (5 CP)			
Prüfungen	mind. 13 CP benötigt					
Mathematik / Logik	Mathematik 1 (8 CP)	Mathematik 2 (8 CP)				
		Logik (5 CP)				
Prüfungen		mind. 10 CP benötigt				
Mathematik / Theoretische Informatik		Mathematik 3 (6 CP)				
		Grundlagen der Theo. Informatik (5 CP)	Grundlagen der Theo. Informatik 2 (5 CP)			
Prüfungen		mind. 10 CP benötigt				
Nebenfach		Nebenfach (5 CP)		Nebenfach (5 CP)		Nebenfach (5 CP)
Prüfungen					mind. 14 CP benötigt	
	5 CP	5 CP				

Lernziele für heute

- ich habe einen Überblick über Inhalte und Organisation der Veranstaltung
- ich habe einen Überblick über die genutzte Software
- ich habe ein intuitives Verständnis dafür, was logisches Schlussfolgern ist

Literatur:

Jon Barwise und John Etchemendy: *Sprache, Beweis und Logik.*
Mentis-Verlag, 2005. Abschnitte Einführung, 2.1, 2.5

Motivation
ooooo

Inhalte
●ooo

Organisation
oooooo

logische Folgerungen
oooooooooooooooooooo

Zusammenfassung
oo

Inhalte der Veranstaltung

Über die Veranstaltung

- Aussagenlogik
 - Syntax und Semantik der Aussagenlogik
 - illustriert mittels einer Klötzchenwelt (Bivalence World)
 - Aussagenlogische Folgerung
 - Aussagenlogische Beweise mit Fitch
 - Aussagenlogische Erfüllbarkeit: DPLL, SAT-Solver
 - Korrektheit, Vollständigkeit
 - Dreiwertige Logik
- Prädikatenlogik erster Stufe (PL1)
 - Allquantoren, Existenzquantoren
 - Prädikatenlogik universell: PL1-Strukturen
 - logische Folgerungen erster Stufe
 - Beweise erster Stufe mit Fitch
 - Korrektheit, Vollständigkeit
 - Anwendungen, Ausblick

Literatur

Jon Barwise und John Etchemendy: *Sprache, Beweis und Logik.*
Mentis-Verlag, 2005, ISBN 978-3-89785-440-6.

Jon Barwise und John Etchemendy: *Sprache, Beweis und Logik II.*
Mentis-Verlag, Paderborn, 2006, ISBN 978-3-89785-441-3.

Diese beiden Bände sind die deutsche Übersetzung des Buches:

Jon Barwise und John Etchemendy: *Language, Proof and Logic.*
First Edition, Seven Bridges Press, New York, 1999, ISBN
1-889119-08-3.

Software für die Veranstaltung

Boole Erstellen von Wahrheitstabellen

Bivalence World Evaluieren logischer Sätze und Formeln in einer Welt von Klötzchen

Fitch Konstruieren von Beweisen

- Wir Arbeiten mit Version 2 der Software.
- Im Buch wird Tarski's World statt Bivalence World benutzt
 - beide Programme sind sehr ähnlich
 - Bivalence World hat zusätzliche Funktionalitäten.

Motivation
ooooo

Inhalte
oooo

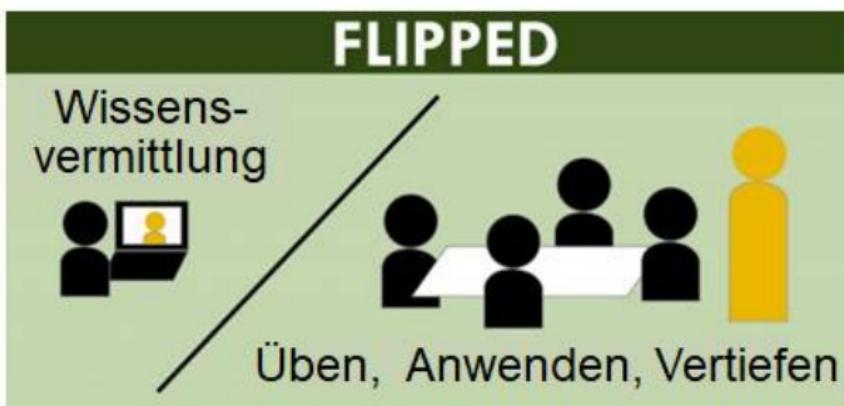
Organisation
●ooooo

logische Folgerungen
ooooooooooooooo

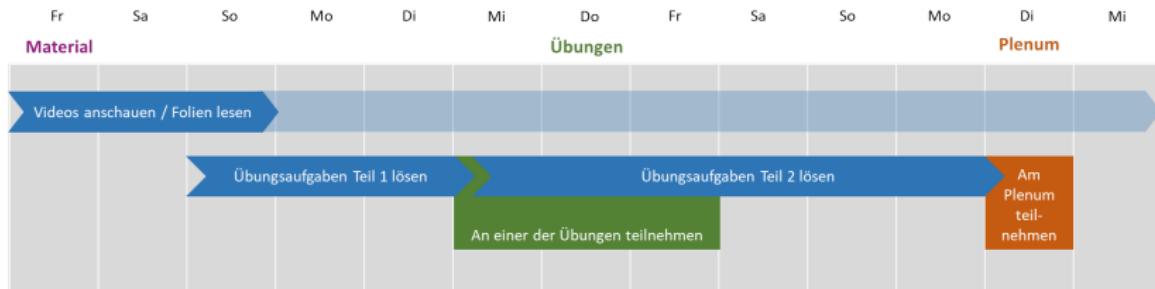
Zusammenfassung
oo

Organisation der Lehrveranstaltung

Flipped classroom

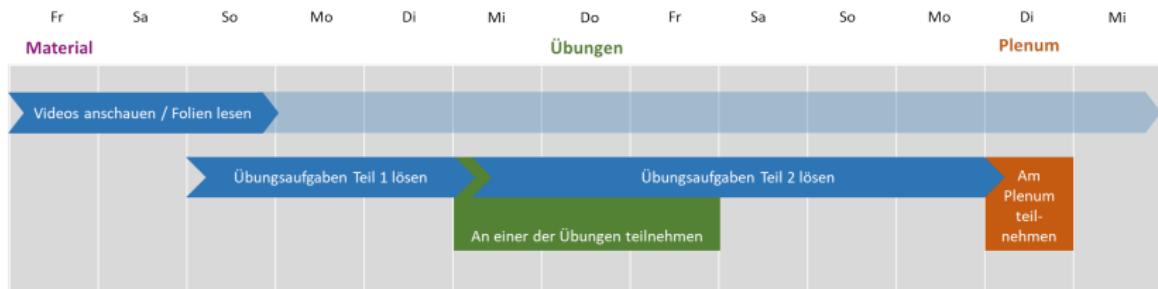


Organisation des flipped classroom



- Die Veranstaltung findet als **flipped classroom** statt
 - Folien werden vor Übung+Plenum ins Moodle gestellt
 - Videos sind auf dem Mediaserver zu finden (Link über Moodle)
 - Sehen Sie sich die Videos vorher an
- zentrale Veranstaltungs-Webseite ist das Moodle:
<https://elearning.ovgu.de>

Organisation: Übungen und Plenum



- **Übungen:** Einschreibung notwendig über Moodle (Fristen siehe dort)
- Pro Woche gibt es ein Übungsblatt mit Aufgaben
- Übungsaufgaben Teil 1 bitte bis zur Übungsgruppe lösen
 - Besprechung mit Votierung in der Übungsgruppe
- Übungsaufgaben Teil 2 werden in der Übungsgruppe gelöst
 - Tutor:innen beantworten Fragen
- **Plenum:**
 - vertiefende Diskussion weiterer Aufgaben und Projekte sowie ggf. der Lösungen der Aufgaben Teil 2

Zulassungskriterien für die Klausur

Zulassungskriterien für die Klausur:

- 66% der Teil-1-Aufgaben müssen **votiert** werden, d.h.
 - Bereitschaft zum Vortrag der Lösung ausdrücken
 - ... und ggf. auch die Lösung vorstellen
 - Lösungen müssen nicht perfekt sein
- Aktive Teilnahme an einer Übung (Bearbeitung von Übungsaufgaben Teil 2).

Damit Sie die Klausur (gut) bestehen können, sollten Sie sich **selbstverantwortlich** darauf vorbereiten:

- Videos und Folien ansehen
- Übungsaufgaben lösen
- Teilnahme an den **Übungen** und am **Plenum**

Zeitaufwand

Der **Zeitaufwand** für die Logik-Veranstaltung beträgt pro Woche:

- ca. 2 Stunden Vorbereitung mittels Video und Übungsaufgaben, Teil 1
- 1,5 Stunden Mitarbeit in der Übung
- ca. 3 Stunden für Lösung der Übungsaufgaben, Teil 2
- 1,5 Stunden Mitarbeit im Plenum

10 Stunden/Woche * 14 Wochen	140 Stunden
Klausur-Vorbereitung	8 Stunden
Klausur	2 Stunden
Summe	150 Stunden (5 CP)

Motivation
ooooo

Inhalte
oooo

Organisation
oooooo

logische Folgerungen
●ooooooooooooooo

Zusammenfassung
oo

Logische Folgerungen in natürlicher Sprache

Aussagen

Definition

Eine **Aussage** ist ein sprachliches oder gedankliches Gebilde, dem genau einer der beiden Wahrheitswerte wahr oder falsch zukommt.

- Wir gehen von einer **zweiwertigen** Logik aus.
- Wir werden natürlichsprachliche Aussagen später als **Sätze** formalisieren.

Logische Argumente

Definition

Ein (logisches) **Argument** besteht aus

- einer Folge von Aussagen, den **Prämissen**, und
- einer Aussage, der **Konklusion**.

Das Argument behauptet, dass die Konklusion aus den Prämissen logisch folgt.

Alle Menschen sind sterblich. Sokrates ist ein Mensch. Also ist Sokrates sterblich.

Alle Menschen sind sterblich. Sokrates ist ein Mensch. Also ist Sokrates sterblich.

Lucretius ist ein Mensch. Schließlich ist Lucretius sterblich, und alle Menschen sind sterblich.

Lucretius ist ein Mensch. Schließlich ist Lucretius sterblich, und alle Menschen sind sterblich.

Fitch-Notation für Argumente

Alle Menschen sind sterblich.
Sokrates ist ein Mensch.
Sokrates ist sterblich.

Prämissen₁
:
Prämissen_n
Konklusion

Finde die Argumente

Wo ist Lea?

Entweder spielt sie draussen Fußball oder liest in ihrem Zimmer.

Es regnet.

Na, dann liest sie wohl.

Es regnet

Wenn es regnet, spielt Lea kein Fußball

Lea spielt kein Fußball

Lea spielt kein Fußball

Lea spielt Fußball oder liest

Lea liest

Gültige logische Argumente

Ein logisches Argument ist **gültig** (oder eine **logische Folgerung**), wenn unter allen Umständen, unter denen die Prämissen wahr sind, auch die Konklusion wahr ist.

Ein gültiges Argument mit wahren Prämissen heißt **korrekt** (oder auch **beweiskräftig, schlüssig**).

(Un)gültige und (in)korrekte logische Argumente

- Alle Menschen sind sterblich. Sokrates ist ein Mensch. Also ist Sokrates sterblich. (**gültig, korrekt**)
- Alle reichen Schauspieler sind gute Schauspieler. Brad Pitt ist ein reicher Schauspieler. Also ist er ein guter Schauspieler. (**gültig, aber nicht korrekt**)
- Alle reichen Schauspieler sind gute Schauspieler. Brad Pitt ist ein guter Schauspieler. Also ist er ein reicher Schauspieler. (**nicht gültig**)

Wie zeigt man die (Un-)Gültigkeit von Argumenten?

Gültigkeit Um zu zeigen, dass ein Argument **gültig** ist, müssen wir über alle Welten argumentieren, und zeigen, dass in jeder Welt die Wahrheit von den Prämissen auf die Konklusion übertragen wird.

(Später werden wir solche Argumentationen präzisieren als **Beweise**.)

Ungültigkeit Ein Argument kann als **ungültig** nachgewiesen werden, indem man ein Gegenbeispiel findet, d. h. eine Welt, in der alle Prämissen wahr sind, aber die Konklusion falsch ist.

Ein Gegenbeispiel

Folgendes Argument ist nicht gültig:

Alle reichen Schauspieler sind gute Schauspieler. Brad Pitt ist ein guter Schauspieler.

Also ist er ein reicher Schauspieler.

Dies können wir zeigen durch folgendes [Gegenbeispiel](#):

Stellen wir uns eine Welt vor, in der

- *alle reichen Schauspieler gute Schauspieler sind,*
- *Brad Pitt ein guter Schauspieler ist, und*
- *Brad Pitt ein armer Schauspieler ist.*

In dieser Welt sind die beiden Prämissen des Arguments wahr, die Konklusion aber falsch, also ist die betrachtete Welt ein Gegenbeispiel gegen die Gültigkeit des obigen Arguments.

Was folgt?

Betrachten wir folgende Prämissen:

Wenn es regnet, dann ist die Straße nass.

Es regnet.

Was folgt daraus als Konklusion?

- Die Straße ist nass?

Folgt, denn es regnet laut zweiter Prämissen, und in diesem Fall folgt laut erster Prämissen, dass die Straße nass.

Was folgt?

Betrachten wir folgende Prämissen:

Wenn es regnet, dann ist die Straße nass.

Die Straße ist nass.

Was folgt daraus als Konklusion?

- Es regnet?

Folgt nicht. Gegenbeispiel: eine Situation, in der es nicht regnet, die Straße aber trotzdem nass ist (z.B. weil jemand Wasser verschüttet hat).

Was folgt?

Betrachten wir folgende Prämissen:

In den letzten Jahrtausenden hat der Mond der Erde immer die gleiche Seite gezeigt.

Was folgt daraus als Konklusion?

- Auch in Zukunft wird der Mond der Erde immer die gleiche Seite zeigen.

Folgt nicht. Gegenbeispiel: eine Situation, in der Mond irgendwann damit aufhört, z.B. wegen Einschlags eines Kometen oder durch große Energiezufuhr durch Außerirdische. Oder die physikalischen Gesetze ändern sich.

Was folgt?

Betrachten wir folgende Prämissen:

Alle Collies sind Hunde

Rover ist ein Hund

Was folgt daraus als Konklusion?

- Rover ist ein Collie?

Folgt nicht. Gegenbeispiel: Rover ist ein Pudel.

Was folgt?

Betrachten wir folgende Prämissen:

Ein verletzter Hund liegt auf der Straße.

Ein beschädigtes Auto steht neben dem Hund.

Was folgt daraus als Konklusion?

- Das Auto hat den Hund verletzt?

Folgt nicht. Gegenbeispiel: der Hund wurde von einem Fußgänger beim Überqueren der Straße verletzt, und das Auto (das vorige Woche durch einen Unfall beschädigt wurde) hat angehalten, um zu helfen.

Indizienbeweise sind also nicht notwendigerweise logische Folgerungen.

Was folgt?

Betrachten wir folgende Prämissen:

Albert Einstein ist ein großer Physiker.

Albert Einstein sagt: die Quantentheorie ist falsch.

Was folgt daraus als Konklusion?

- Die Quantentheorie ist falsch.

Folgt nicht. Gegenbeispiel: eine Welt, in der die Quantentheorie richtig ist, aber Einstein trotz seiner falschen Aussage über die Quantentheorie ein großer Physiker ist.

Für die Frage von logischer Folgerung zählen Autoritäten nichts.

Was folgt?

Betrachten wir folgendes Programm P:

```
while x<4 {  
    x = x+1;  
    y = y+1;  
}
```

und betrachten wir folgende Prämisse:

x hat vor Ausführung von P den Wert 0.

Was folgt daraus als Konklusion?

- y hat nach Ausführung von P einen um 4 höheren Wert als vor der Ausführung.

Folgt, weil die while-Schleife genau viermal durchlaufen wird.

Motivation
ooooo

Inhalte
oooo

Organisation
oooooo

logische Folgerungen
oooooooooooooooooooo

Zusammenfassung
●○

Zusammenfassung

Zusammenfassung und Ausblick

Was haben wir heute gelernt?

- was sind logische Argumente?
- wann sind diese gültig? korrekt?
- wie kann man entscheiden, ob ein Argument gültig ist oder nicht?

Wie geht es weiter?

- nächste Woche: Syntax logischer Sätze und Formalisierung
- in zwei Wochen: Semantik logischer Sätze
- in drei Wochen: logische Folgerung, präzise definiert

Lernziele 2

●○

künstliche Sprache

○○○○

Atomare Sätze

○○○○○○○○○○○○○○○○

Komplexe Sätze

○○○○○○○○○○○○○○

Zusammenfassung

○○

Lernziele Veranstaltung 2

Lernziele für heute

- ich verstehe, warum **formale Sprachen** notwendig sind
- ich bin in der Lage, einfache deutsche Sätze in logische Sätze (Prädikatenlogik) zu übersetzen (**formalisieren**)
- ich kann das Programm **Bivalence World** bedienen
- ich kenne die **Syntax der Aussagenlogik**
- ich kenne die **Semantik atomarer Sätze**

Buch: Einführung, 1.1–1.5, 1.7, 3.1–3.3, 3.7, 7.1, 7.2

Warum eine künstliche Sprache?

Mehrdeutigkeit natürlicher Sprache

Überlegen Sie sich: Was bedeutet folgender Satz?

Ich sah den Mann auf dem Berg mit dem Fernrohr

Ich sah den Mann auf dem Berg mit dem Fernrohr



((((Ich sah den Mann) auf dem Berg) mit dem Fernrohr))



((Ich sah (den Mann auf dem Berg)) mit dem Fernrohr))



((Ich sah den Mann) (auf dem Berg mit dem Fernrohr))



(Ich sah ((den Mann auf dem Berg) mit dem Fernrohr))



(Ich sah (den Mann (auf dem Berg mit dem Fernrohr)))

5 mögliche Interpretationen!

Warum eine künstliche (formale) Sprache?

- Studium von Prinzipien des korrekten Schließens:
 - Was ist eine logische Folgerung?
 - Erfordert das Eliminieren jedweder Mehrdeutigkeit.
 - Aufdecken von Mehrdeutigkeiten in natürlichen Sprachen.
 - Ähnliche wie bei Programmiersprachen: diese sind auch formale Sprachen.

Den Übergang von einer natürlichen zu einer formalen Sprache nennt man **Formalisierung**.

Lernziele 2

○○

künstliche Sprache

○○○○

Atomare Sätze

●○○○○○○○○○○○○○○○○

Komplexe Sätze

○○○○○○○○○○○○○○○○

Zusammenfassung

○○

Atomare Sätze in PL1

Die Sprache PL1: Individuenkonstanten

Definition

Eine **Individuenkonstante** ist ein Symbol, das genau eine Person, Ding oder Objekt bezeichnet.

- Beispiele:
 - Zahlen: 0, 1, 2, 3, ...
 - Namen: max, claire
 - Formale Konstanten: a, b, c, d, e, f, n1, n2
 - Jede Individuenkonstante muss ein existierendes Objekt bezeichnen.
 - Keine Individuenkonstante kann mehr als ein Objekt bezeichnen.
 - Ein Objekt kann keinen, einen oder mehrere Namen (Individuenkonstanten) haben.

Die Sprache PL1: Prädikatsymbole

Definition

Ein **Prädikatsymbol** Pr ist ein Symbol, das eine Eigenschaft eines Objektes oder einer Beziehung (Relation) zwischen Objekten bezeichnet.

Jedes Prädikatsymbol hat eine **Stelligkeit**, die die Anzahl der Objekte angibt, die in Beziehung stehen.

- Beispiele:

- Stelligkeit 0: Gate0_is_low, A, B, ...
- Stelligkeit 1:
Cube (Würfel), Tet (Tetraeder), Dodec (Dodekaeder),
Small (Klein), Medium (Mittelgroß), Large (Groß)
- Stelligkeit 2:
Smaller (Kleiner), Larger (Größer),
LeftOf (Links von), BackOf (Hinter),
Adjoins (Benachbart), SameSize (Gleichgroß), ...
- Stelligkeit 3: Between (Zwischen)

Die Interpretation von Prädikatsymbolen

- In **Bivalence World** besitzt jedes Prädikatsymbol eine feste Bedeutung, die meist mit der Interpretation in einer natürlichen Sprache übereinstimmt.
- In anderen PL1-Sprachen kann die Interpretation von Prädikatsymbolen variieren. Zum Beispiel kann \leq eine Ordnung von Zahlen, von Zeichenketten, Bäumen, etc. bedeuten.
- Wir vereinbaren, dass das Symbol „=“ immer ein zweistelliges Prädikatensymbol ist, und **Gleichheit (Identität)** bedeutet.

Prädikatsymbole in Bivalence World

Atomarer

Satz

Interpretation

Tet(a)	a ist ein Tetraeder
Cube(a)	a ist ein Würfel
Dodec(a)	a ist ein Dodekaeder
Small(a)	a ist klein
Medium(a)	a ist mittelgroß
Large(a)	a ist groß
SameSize(a,b)	a ist genauso groß wie b
SameShape(a,b)	a hat dieselbe Form wie b
Larger(a,b)	a ist größer als b
Smaller(a,b)	a ist kleiner als b
SameCol(a,b)	a ist in derselben Spalte wie b
SameRow(a,b)	a ist in derselben Zeile wie b
Adjoins(a,b)	a und b befinden sich auf (nicht diagonal) benachbarten Quadranten
LeftOf(a,b)	a befindet sich dem linken Rand des Feldes näher als b
RightOf(a,b)	a befindet sich dem rechten Rand des Feldes näher als b
FrontOf(a,b)	a befindet sich der Vorderkante des Feldes näher als b
BackOf(a,b)	a befindet sich der Hinterkante des Feldes näher als b
Between(a,b,c)	a, b und c befinden sich in derselben Zeile, Spalte oder Diagonale, und a liegt zwischen b und c

Syntax und Semantik atomarer Sätze (vorläufige Definition)

Syntax: Ein **atomarer Satz** ist eine Anwendung eines n -stelligen Prädikatsymbols Pr auf n Individuenkonstanten c_1, \dots, c_n :

$$\text{Pr}(c_1, \dots, c_n)$$

Manche zweistellige Prädikatsymbole wie $=$ und \leq werden zwischen die Argumente platziert. Statt $= (a, b)$ schreiben wir:

$$a = b$$

Semantik: Ein atomarer Satz ist in einer Welt (z.B. einer Bivalence World-Welt) **wahr**, wenn die Eigenschaft oder Relation, die Pr bezeichnet, auf die Objekte zutrifft, die durch die Individuenkonstanten c_1, \dots, c_n bezeichnet werden (sonst ist er **falsch**).

In Bivalence World ist die Interpretation der Prädikatsymbole **fest** (siehe vorige Folie). Die Interpretation der Konstanten ist hingegen **flexibel**: sie kann in jeder Welt neu festgelegt werden.

Beispiele atomarer Sätze

Kurzfassung der Definition (vorige Folie):

atomarer Satz:

Prädikatsymbol, angewandt auf Individuenkonstanten

Beispiele atomarer Sätze:

- mit einem einstelligen Prädikatsymbol: $\text{Dodec}(a)$
- mit einem zweistelligen Prädikatsymbol: $\text{Larger}(a,b)$
 - Reihenfolge der Argumente ist wichtig:
 $\text{Larger}(a,b)$ vs. $\text{Larger}(b,a)$
 - $= (a,b)$ wird als $a=b$ notiert; ebenso für $<$ u.a.
- mit einem nullstelligen Prädikatsymbol: A
 - bei nullstelligen Prädikatsymbolen werden gibt es keine (null) Argumente; daher können auch die Klammern weggelassen werden

Formalisierung atomarer Sätze

- Bei dem mit a bezeichnete Objekt handelt es sich um einen Würfel. $\text{Cube}(a)$
- a ist kleiner als b $\text{Smaller}(a,b)$
- a ist genauso groß wie es selbst $\text{SameSize}(a,a)$
- Zwischen a und b befindet sich c $\text{Between}(c,a,b)$
- a und b bezeichnen dasselbe Objekt $a=b$
- Das mit a bezeichnete Objekt ist identisch zu sich selbst
 $a=a$

Motivation: Funktionssymbole und Terme

Individuenkonstanten bezeichnen Objekte

- Beispiel aus dem Bereich der Zahlen: 0, 1, 2, 3, 4

Mit **Funktionssymbolen** können Objekte in komplexer Weise notiert werden:

- 3+4
- -2 * (3 + 4)

Die Funktionssymbole sind hier +, -, *

Die Sprache PL1: Funktionssymbole

Bisher können Objekte nur mit Individuenkonstanten bezeichnet werden. **Funktionssymbole** und daraus gebildete **Terme** erlauben, Objekte in komplexerer und mächtigerer Weise zu bezeichnen.

Definition

Eine **Funktionssymbol** f ist ein Symbol, das eine Abbildung (Funktion) zwischen Objekten bezeichnet.

Jedes Funktionssymbol hat eine **Stelligkeit** $n > 0$, die die Anzahl der benötigten Argumente angibt.

Sowohl Prädikate als auch Funktionen werden auf Objekte **angewandt**. Das **Resultat**

- einer Funktion ist wiederum ein **Objekt**.
- eines Prädikats ist ein **Wahrheitwert**.

Beispiele für Funktionssymbole

- Arithmetik (einstellig): $-$
- Arithmetik (zweistellig): $+$, $-$, $*$, $/$
- in Bivalence World (einstellige Funktionssymbole):
 rm , lm , fm , bm . Interpretation:
 - $rm(a)$: (“rightmost”)
das am weitesten rechts stehende Objekt in der Zeile von a
 - $lm(a)$: (“leftmost”)
das am weitesten links stehende Objekt in der Zeile von a
 - $fm(a)$: (“frontmost”)
das am weitesten vorne stehende Objekt in der Spalte von a
 - $bm(a)$: (“backmost”)
das am weitesten hinten stehende Objekt in der Spalte von a
- Verwandtschaftsbeziehungen (einstellig): $father$, $mother$

Die Sprache PL1: variablenfreie Terme

Definition

Variablenfreie Terme sind induktiv wie folgt definiert:

- jede Individuenkonstante c ist ein Term,
- wenn t_1, \dots, t_n Terme sind und f ein n -stelliges Funktionssymbol, dann ist auch $f(t_1, \dots, t_n)$ ein Term.

Kurzschreibweise für diese Definition:

$$\begin{array}{ll} t ::= c & \text{Basisfall: Individuenkonstanten} \\ | \ f^{(n)}(t_1, \dots, t_n) & \text{Rekursiver Fall} \end{array}$$

Damit ergeben sich z. B. die Terme

- in Bivalence World: $\text{rm}(a)$, $\text{bm}(\text{rm}(a))$
- $\text{father}(\text{max})$, $\text{mother}(\text{father}(\text{max}))$
- $3*(4+2)$

Das Funktionssymbol steht hier zwischen den Argumenten

Syntax und Semantik atomarer Sätze (revidierte Definition)

Definition (Syntax)

Ein **atomarer Satz** ist eine Anwendung eines n -stelligen Prädikatsymbols Pr auf n variablenfreie Terme t_1, \dots, t_n :

$$\text{Pr}(t_1, \dots, t_n)$$

- Damit lassen sich z. B. neue atomare Sätze bilden:

- in Bivalence World: $\text{SameCol}(\text{a}, \text{fm}(\text{a}))$
- $\text{Larger}(\text{father}(\text{max}), \text{max})$
- $2 < 3 * (4 + 2), -(2 * 3) = -6$

Semantik: Ein atomarer Satz ist in einer Welt (z.B. einer Bivalence World-Welt) **wahr**, wenn die Eigenschaft oder Relation, die Pr bezeichnet, auf die Objekte zutrifft, die durch die Terme t_1, \dots, t_n bezeichnet werden (sonst ist er **falsch**).

Formalisierung atomarer Sätze mit Funktionssymbolen

Wir verwenden nur die auf der vorigen Folie eingeführten Funktionssymbole.

- das von a aus am weitesten rechts stehende Objekt befindet sich in der selben Spalte wie b $\text{SameCol}(\text{rm}(a), b)$
- b steht von a aus am weitesten rechts $\text{rm}(a) = b$
- a ist das einzige Objekt in seiner Zeile $\text{Im}(a) = \text{rm}(a)$
- 4 plus 5 ist größer als 7 $4 + 5 > 7$
- Peter ist jünger als der Großvater väterlicherseits von Max
 $\text{Jünger}(\text{peter}, \text{father}(\text{father}(\text{max})))$
- Max und Laura haben den gleichen Vater
 $\text{father}(\text{max}) = \text{father}(\text{laura})$

Sprache erster Stufe

Definition

Eine Sprache erster Stufe besteht aus

- einer Menge von Prädikatsymbolen mit Stelligkeiten, wie z. B. *Smaller*⁽²⁾, *Dodec*⁽¹⁾, *Between*⁽³⁾, \leq ⁽²⁾, einschließlich der aussagenlogischen Symbole (nullstellige Prädikatsymbole), wie z. B. $A^{(0)}$, $B^{(0)}$, $C^{(0)}$, (großgeschrieben),
 - das Gleichheitssymbol = ist stets ein zweistelliges Prädikatsymbol,
- einer Menge von Namen oder Individuenkonstanten, wie z. B. a , b , c , (kleingeschrieben),
- einer Menge von Funktionssymbolen mit Stelligkeiten, wie z. B. $f^{(1)}$, $^{+(2)}$, $^{×(2)}$ (kleingeschrieben).

Üblicherweise werden die Stelligkeiten weggelassen.

Atomare Sätze (und später auch komplexe Sätze) sind immer relativ zu einer Sprache erster Stufe definiert.

Komplexe aussagenlogische Sätze

Syntax aussagenlogischer Sätze

Die Ausdrucksmächtigkeit atomarer Sätze ist begrenzt. In Bivalence World können wir folgende Aussagen nicht mit atomaren Sätzen ausdrücken:

- a ist kein Würfel, d.h. es ist **nicht** der Fall, dass a ein Würfel ist.
- a ist ein großer Würfel, d.h. a ist ein Würfel **und** a ist groß.
- a ist ein Würfel **oder** ein Dodekaeder.
- **wenn** a ein Würfel ist, **dann** ist a auch groß.
- a ist ein Würfel **genau dann wenn** a groß ist.

Syntax aussagenlogischer (quantorenfreier) PL1-Sätze

Definition

Gegeben eine Menge \mathcal{A} von atomaren Sätzen, definieren wir die Menge der **Sätze** über \mathcal{A} induktiv wie folgt:

- ① jeder atomare Satz aus \mathcal{A} ist ein aussagenlogischer Satz;
- ② \perp (Widerspruch) ist ein aussagenlogischer Satz;
- ③ ist P ein aussagenlogischer Satz, dann auch $\neg P$ (Negation);
- ④ wenn P_1 und P_2 aussagenlogische Sätze sind, dann auch $(P_1 \vee P_2)$ (Disjunktion), $(P_1 \wedge P_2)$ (Konjunktion), $(P_1 \rightarrow P_2)$ (Implikation) sowie $(P_1 \leftrightarrow P_2)$ (Biimplikation).

Kurz: $P ::= A \mid \perp \mid \neg P \mid (P_1 \vee P_2) \mid (P_1 \wedge P_2) \mid (P_1 \rightarrow P_2) \mid (P_1 \leftrightarrow P_2)$

Wir verwenden als \mathcal{A} die atomaren Sätze im Sinne von Folie 54.

Diese sind relativ zu einer PL1-Sprache definiert.

Insofern sind auch Sätze relativ zu einer PL1-Sprache definiert.

Aussagenlogische PL1-Sätze: Beispiele

Cube(a)	a ist ein Würfel
$\neg \text{Cube}(a)$	Es nicht der Fall, dass a ist ein Würfel ist (d.h. a ist kein Würfel)
$\neg\neg \text{Cube}(a)$	Es nicht der Fall, dass es nicht der Fall ist, dass a ein Würfel ist (d.h. a ist nicht kein Würfel)
$\neg\neg\neg \text{Cube}(a)$	Es nicht der Fall, dass es nicht der Fall ist, dass es nicht der Fall ist, dass a ein Würfel ist (d.h. a ist nicht nicht kein Würfel)

Aussagenlogische PL1-Sätze: Beispiele

$(\text{Cube}(a) \wedge \text{Large}(a))$	a ist ein großer Würfel
$(\text{Cube}(a) \vee \text{Large}(a))$	a ist ein Würfel oder groß
$(\text{Cube}(a) \rightarrow \text{Large}(a))$	wenn a ein Würfel ist, ist es groß
$(\text{Cube}(a) \leftrightarrow \text{Large}(a))$	a ist ein Würfel genau dann, wenn es groß ist

Aussagenlogische PL1-Sätze: Beispiele

$(\text{Cube}(a) \rightarrow (\text{Large}(a) \vee \text{Medium}(a)))$	Wenn a ein Würfel ist, ist a groß oder mittel
$(\text{Cube}(a) \leftrightarrow \neg \text{Large}(a))$	a ist ein Würfel genau dann, wenn es nicht groß ist
$(\text{Cube}(a) \rightarrow (\text{Large}(a) \rightarrow \text{LeftOf}(a,b)))$	Wenn a ein Würfel ist, dann ist es der Fall, dass wenn a groß ist, dann liegt es links von b

Aussagenlogische PL1-Sätze: Klammerkonventionen

Wir vereinbaren folgendene Klammerkonventionen:

- Die äußeren Klammern können weggelassen werden.
- Zusätzliche Klammern dürfen eingeführt werden.

Schreibweise laut Definition	mit Klammerkonvention
$\neg \text{Cube}(a)$	$\neg \text{Cube}(a)$
$\neg\neg \text{Cube}(a)$	$\neg\neg \text{Cube}(a)$
$(\text{Cube}(a) \wedge \text{Large}(a))$	$\text{Cube}(a) \wedge \text{Large}(a)$
$(\text{Cube}(a) \rightarrow (\text{Large}(a) \vee \text{Small}(a)))$	$\text{Cube}(a) \rightarrow (\text{Large}(a) \vee \text{Small}(a))$
$(\text{Cube}(a) \leftrightarrow \neg \text{Large}(a))$	$\text{Cube}(a) \leftrightarrow (\neg \text{Large}(a))$

Formalisierung komplexer Sätze: Negationen

- Negationen werden oft durch “kein”, “un-” u.a. ausgedrückt
 - a ist kein Tetraeder
 $\neg\text{Tet}(a)$
 - 9 ist ungerade
 $\neg\text{Gerade}(9)$

Formalisierung komplexer Sätze: Konjunktionen

- Konjunktionen verbinden **Sätze**, und **nicht Objekte**:
 - a und b sind Würfel $\text{Cube}(a) \wedge \text{Cube}(b)$
falsche Notation: ~~$\text{Cube}(a \wedge b)$~~
- Der deutsche Ausdruck **und** legt manchmal eine **zeitliche Ordnung** nahe, der PL1-Ausdruck \wedge tut dies nie.
 - Peter drückte auf den Schalter und das Licht ging an
 $\text{Drückte}(\text{peter}, \text{schalter}) \wedge \text{GingAn}(\text{licht})$
- Die deutschen Ausdrücke **aber**, **jedoch**, **hingegen**, **dennoch** und **außerdem** sind alle stilistische Varianten von **und**.
 - Es regnet, aber die Straße bleibt trocken
 $\text{Regnet} \wedge \text{Trocken}(\text{straße})$

Formalisierung komplexer Sätze: Disjunktionen

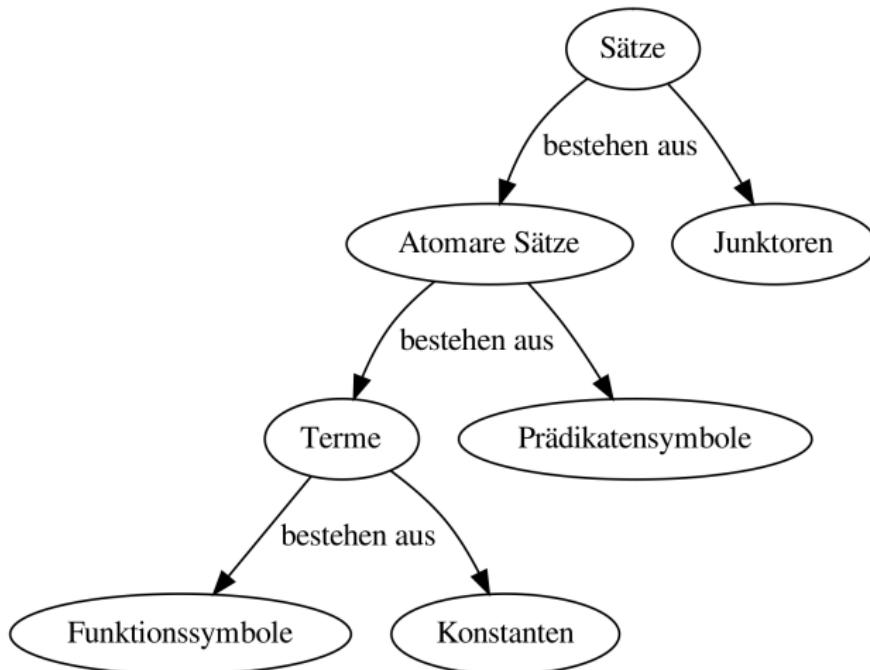
- Das **oder** in der deutschen Sprache kann das **inklusive oder** (\vee) oder das **exklusive oder** (entweder ... oder) bedeuten:
 $A \text{ xor } B \Leftrightarrow (A \vee B) \wedge (\neg A \vee \neg B)$
 - Ich esse entweder Eis oder Schokolade
 $(\text{Esse(eis)} \vee \text{Esse(schokolade)})$
 $\wedge (\neg \text{Esse(eis)} \vee \neg \text{Esse(schokolade)})$
- Die deutsche Wendung **sowohl ... als auch** wird bisweilen wie Klammern verwendet, um einen andernfalls mehrdeutigen Satz zu klären.

Formalisierung von Konditionalsätzen

- Die folgenden deutschen Ausdrücke werden alle als $P \rightarrow Q$ übersetzt:
 - Wenn P , dann Q .
 - Q wenn P .
 - P nur dann, wenn Q .
 - Gegeben dass P , dann Q .
 - P impliziert Q .
- a ist ein Würfel nur dann, wenn es groß ist
 $\text{Cube}(a) \rightarrow \text{Large}(a)$
- Die folgenden Sätze werden als $\neg P \rightarrow Q$ übersetzt:
 - Sofern nicht P , Q .
 - Q , es sei denn dass P .
- a ist ein Würfel, es sei denn, es ist groß
 $\neg \text{Large}(a) \rightarrow \text{Cube}(a)$

Syntax von Sätzen

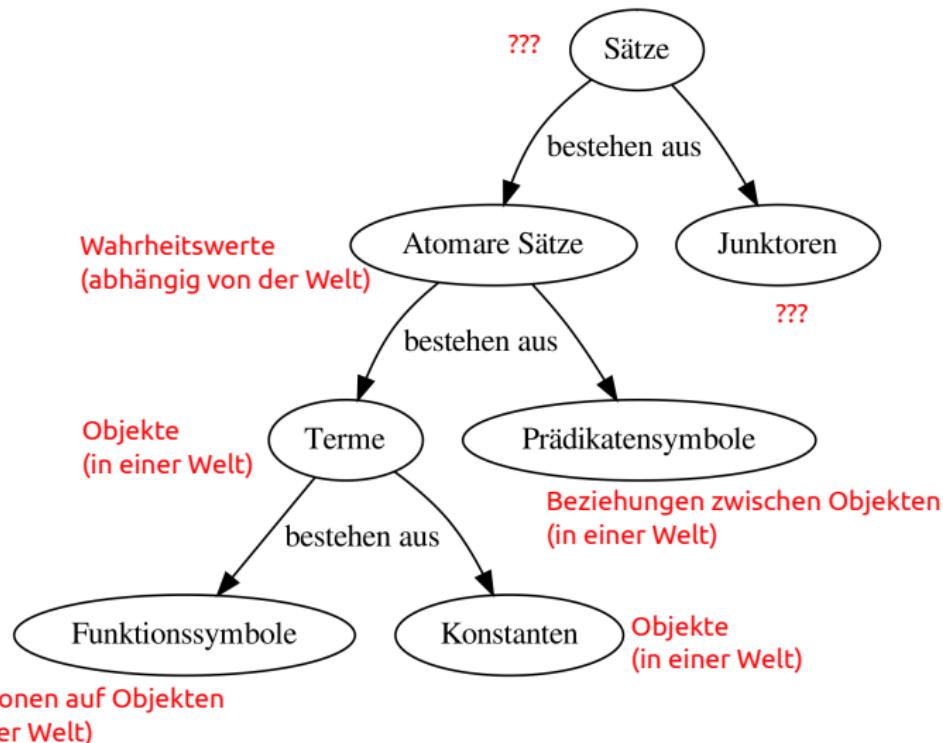
Syntax



Syntax und Semantik von Sätzen

Semantik

Syntax



Lernziele 2

○○

künstliche Sprache

○○○○

Atomare Sätze

○○○○○○○○○○○○○○○○

Komplexe Sätze

○○○○○○○○○○○○○○○○

Zusammenfassung

●○

Zusammenfassung

Zusammenfassung und Ausblick

Was haben wir heute gelernt?

- die Notwendigkeit **formaler Sprachen**
- den Prozess der **Formalisierung**
- atomare Sätze in PL1, ihre Bestandteile und ihre Semantik
- komplexe Sätze in PL1 und ihre Bestandteile

Nächstes Mal behandeln wir:

- **Semantik** der Aussagenlogik
- die **Wahrheitstafelmethode**
- logische **Wahrheit** und **Erfüllbarkeit**

Lernziele Veranstaltung

3

Lernziele für heute (Veranstaltung 3)

- ich verstehe die **Semantik** aussagenlogischer Sätze
- ich verstehe den Unterschied zwischen Syntax und Semantik
- ich kann Sätze beurteilen, ob sie **erfüllbar** und/oder **logisch wahr** sind
- ich verstehe den Unterschied zwischen **Aussagenlogik** und **Bivalence World**
- ich kann die **Wahrheitstafelmethode** und das Programm **Boole** anwenden
 - Sinn: Evaluierung von Sätzen, Überprüfung von logischen Eigenschaften von Sätzen

Buch: 3.1–3.3, 4.1, 7.1, 7.2, 17.1 (letzteres in Band 2)

Semantik der Aussagenlogik

Wiederholung: Aussagenlogische Sätze

Definition

Gegeben eine Menge \mathcal{A} von atomaren Sätzen, definieren wir die Menge der **aussagenlogischen Sätze** über \mathcal{A} induktiv wie folgt:

- ① jeder atomare Satz aus \mathcal{A} ist ein aussagenlogischer Satz;
- ② \perp ist ein aussagenlogischer Satz;
- ③ wenn P ein aussagenlogischer Satz ist, dann auch $\neg P$;
- ④ wenn P_1 und P_2 aussagenlogische Sätze sind, dann auch $(P_1 \vee P_2)$, $(P_1 \wedge P_2)$, $(P_1 \rightarrow P_2)$ sowie $(P_1 \leftrightarrow P_2)$.

\perp , \neg , \vee , \wedge , \rightarrow , \leftrightarrow heißen dabei **Junktoren**.

Semantik aussagenlogischer Sätze

Wenn eine Belegung atomarer Sätze mit Wahrheitswerten gegeben ist, können wir diese mittels folgender **Wahrheitstafeln** auf komplexe Sätze erweitern:

		P	Q	$P \wedge Q$	P	Q	$P \vee Q$
		T	T	T	T	T	T
		F	F	F	F	F	F
P	$\neg P$	T	F	F	T	F	T
T		T	F	F	T	F	T
F	T	F	T	F	F	T	T
		F	F	F	F	F	F

		P	Q	$P \rightarrow Q$	P	Q	$P \leftrightarrow Q$
		T	T	T	T	T	T
		T	F	F	T	F	F
P	$\neg P$	T	F	F	T	F	F
T		T	F	F	T	F	F
F	T	F	T	T	F	T	F
F	F	F	F	T	F	F	T

\perp bedeutet immer F.

Wahrheitstafel: Beispiel

A	B	$A \leftrightarrow B$	$\neg(A \leftrightarrow B)$
T	T	T	F
T	F	F	T
F	T	F	T
F	F	T	F

Wahrheitstafeln können mit dem Programm [Boole](#) erstellt werden.

Wahrheitstafel: Beispiel 2

A	B	C	$A \vee B$	$(A \vee B) \rightarrow C$
T	T	T	T	T
T	T	F	T	F
T	F	T	T	T
T	F	F	T	F
F	T	T	T	T
F	T	F	T	F
F	F	T	F	T
F	F	F	F	T

Wahrheitswertbelegungen: Motivation

Wie können Wahrheitstabellen **mathematisch präzise** formuliert werden?

Beobachtung:

eine **Zeile** in einer Wahrheitstabelle

entspricht

einer **Funktion**

von den atomaren Sätzen in die Menge $\{T, F\}$ der Wahrheitswerte.

Wahrheitswertbelegung und Bewertungsfunktion

Definition

Eine **Wahrheitswertbelegung** ist eine Funktion h von der Menge der atomaren Sätze einer Sprache in die Menge $\{T, F\}$.

Definition (Formalisierung der Wahrheitstafeln von Folie 76)

Die Wahrheitswertbelegung h kann zur **Bewertungsfunktion** \hat{h} von der Menge aller Sätze in die Menge $\{T, F\}$ **erweitert** werden:

- ① $\hat{h}(Q) = h(Q)$ für atomare Sätze Q ;
- ② $\hat{h}(\perp) = F$;
- ③ $\hat{h}(\neg Q) = T$ genau dann, wenn $\hat{h}(Q) = F$;
- ④ $\hat{h}(Q \wedge R) = T$ genau dann, wenn $\hat{h}(Q) = \hat{h}(R) = T$;
- ⑤ $\hat{h}(Q \vee R) = F$ genau dann, wenn $\hat{h}(Q) = \hat{h}(R) = F$;
- ⑥ $\hat{h}(Q \rightarrow R) = F$ genau dann, wenn $\hat{h}(Q) = T$ und $\hat{h}(R) = F$;
- ⑦ $\hat{h}(Q \leftrightarrow R) = T$ genau dann, wenn $\hat{h}(Q) = \hat{h}(R)$.

Wahrheitstafel: Beispiel in präziser Notation

$h(A)$	$h(B)$	$\hat{h}(A)$	$\hat{h}(B)$	$\hat{h}(A \leftrightarrow B)$	$\hat{h}(\neg(A \leftrightarrow B))$
T	T	T	T	T	F
T	F	T	F	F	T
F	T	F	T	F	T
F	F	F	F	T	F

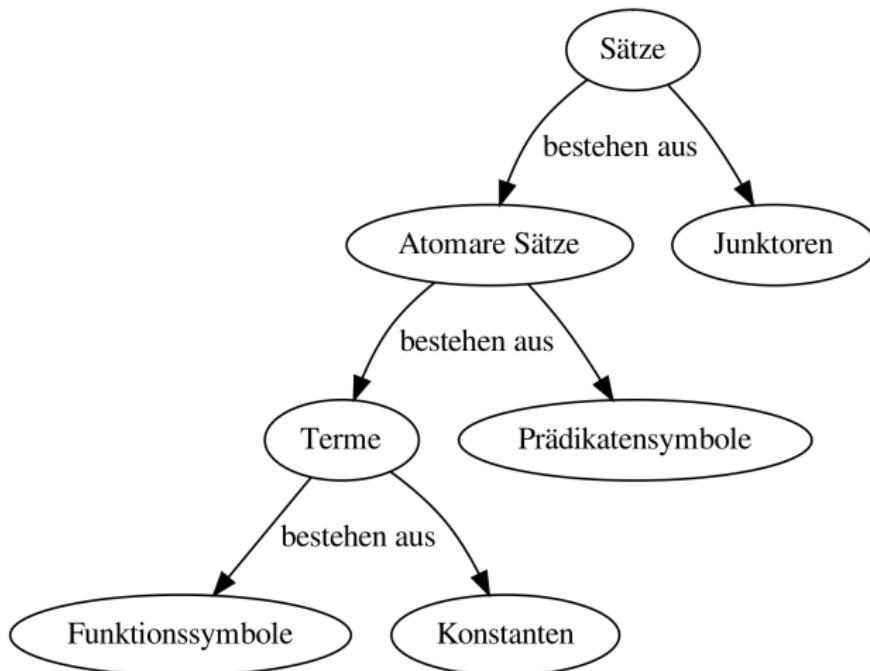
Wahrheitstafel: Beispiel 2 in präziser Notation

$h(A)$	$h(B)$	$h(C)$	$\hat{h}(A)$	$\hat{h}(B)$	$\hat{h}(C)$	$\hat{h}(A \vee B)$	$\hat{h}((A \vee B) \rightarrow C)$
T	T	T	T	T	T	T	T
T	T	F	T	T	F	T	F
T	F	T	T	F	T	T	T
T	F	F	T	F	F	T	F
F	T	T	F	T	T	T	T
F	T	F	F	T	F	T	F
F	F	T	F	F	T	F	T
F	F	F	F	F	F	F	T

Künftig werden wir meist auf die präzise Notation verzichten, und auch die Spalten für $h(A)$ und $\hat{h}(A)$ verschmelzen.

Syntax von Sätzen

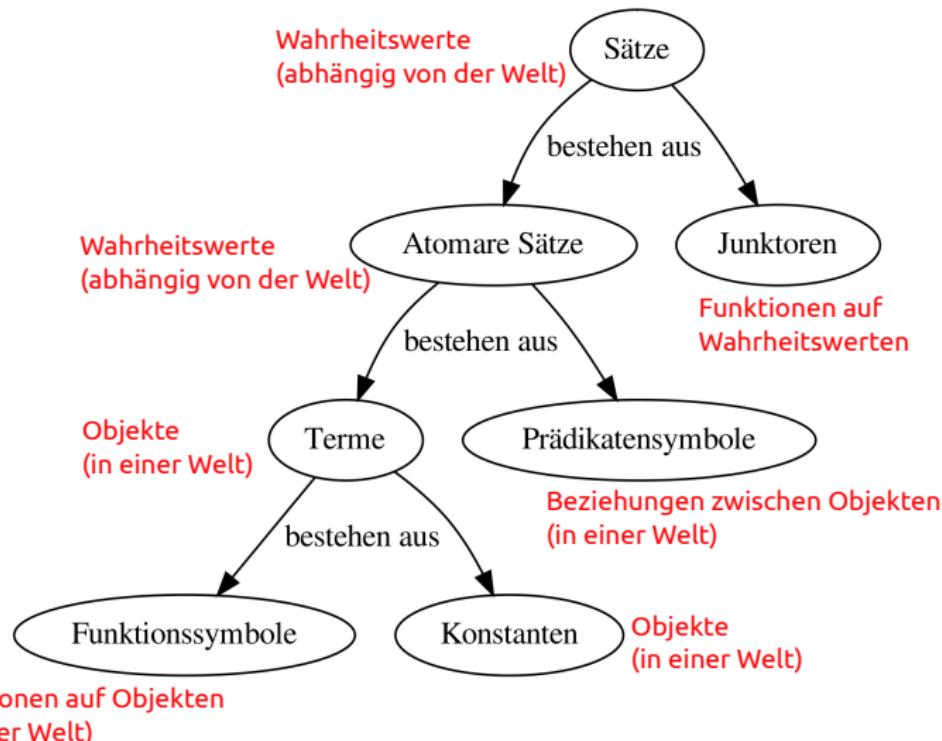
Syntax



Syntax und Semantik von Sätzen

Semantik

Syntax



Wahrheit und Erfüllbarkeit

Logische Wahrheiten

- Logisch notwendige Sätze oder logische Wahrheiten sind Sätze, die unter allen Umständen (in allen Welten) wahr sind.
- Logische Wahrheiten wie $A \vee \neg A$ sind aus rein logischen Gründen wahr.

Definition

Ein Satz P ist Wahrheitstafel-wahr, WT-wahr oder eine Tautologie, wenn er wahr ist für alle Wahrheitswertbelegungen, d.h. für alle Wahrheitswertbelegungen h gilt $\hat{h}(P) = T$.

Logische Wahrheit oder nicht?

- $A \rightarrow A \rightsquigarrow$ ja
- $A \rightarrow B \rightsquigarrow$ nein
- $(A \rightarrow B) \vee (A \rightarrow \neg B) \rightsquigarrow$ ja
- $(A \rightarrow B) \wedge (B \rightarrow A) \rightsquigarrow$ nein
- $(A \rightarrow B) \vee (B \rightarrow A) \rightsquigarrow$ ja
- $A \vee \neg A \rightsquigarrow$ ja
- $(A \vee B) \wedge (\neg A \vee \neg B) \rightsquigarrow$ nein
- $\neg \perp \rightsquigarrow$ ja

Die Wahrheitstafelmethode

- Ein Satz ist genau dann eine **Tautologie**, wenn in allen Zeilen seiner vollständigen Wahrheitstafel der Wahrheitswert T steht.
- Wahrheitstafeln können mit dem Programm **Boole** erstellt werden.

Erfüllbarkeit: Motivation

Mit einer Menge von Sätzen (= **logische Theorie**) können wir

- domänenspezifische **Anforderungen** spezifizieren (z.B. an eine Software oder Produktkonfiguration),
- domänenspezifisches **Wissen** repräsentieren (z.B. mittels einer Ontologie).

Solche Anforderungen oder Wissensbasen sollten
in sich **widerspruchfrei** und **realisierbar** sein.

Daher ist folgender Begriff wichtig:

Logisch mögliche oder erfüllbare logische Theorien sind solche, die unter mindestens einem Umstand (Welt) wahr sind.

Erfüllbarkeit: Definition

Definition

Eine **logische Theorie** ist eine Menge \mathcal{T} von Sätzen (zu einer gegebenen PL1-Sprache).

Definition

Eine logische Theorie \mathcal{T} ist **Wahrheitstafel-erfüllbar** (WT-erfüllbar) wenn für mindestens eine Wahrheitswertbelegung alle Sätze wahr sind, d.h. es gibt eine Wahrheitswertbelegung h mit $\hat{h}(P) = T$ für alle Sätze $P \in \mathcal{T}$. Ansonsten heißt sie **WT-unerfüllbar**.

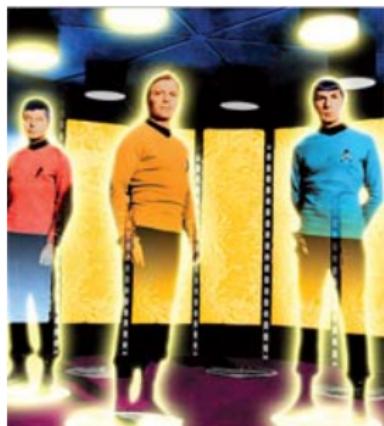
Bemerkung: in der Informatik interessieren uns hauptsächlich **endliche** logische Theorien, d.h. $\mathcal{T} = \{P_1, \dots, P_n\}$.

Obige Bedingung lautet dann: $\hat{h}(P_1) = \dots = \hat{h}(P_n) = T$.

Erfüllbar: ja oder nein

- $\{A, \neg A\} \rightsquigarrow \text{nein}$
- $\{A \rightarrow B\} \rightsquigarrow \text{ja}$
- $\{A \rightarrow B, A \rightarrow \neg B\} \rightsquigarrow \text{ja}$
- $\{A \vee B, \neg A, \neg B\} \rightsquigarrow \text{nein}$
- $\{A \rightarrow \neg A\} \rightsquigarrow \text{ja}$
- $\{A \rightarrow B, A, \neg B\} \rightsquigarrow \text{nein}$
- $\{\perp\} \rightsquigarrow \text{nein}$

Erfüllbar/logisch möglich



Logisch und physikalisch möglich



WT-unmöglich

$$A \wedge \neg A$$

WT-wahr

$$A \vee \neg A$$

Semantik der Aussagenlogik für Bivalence World

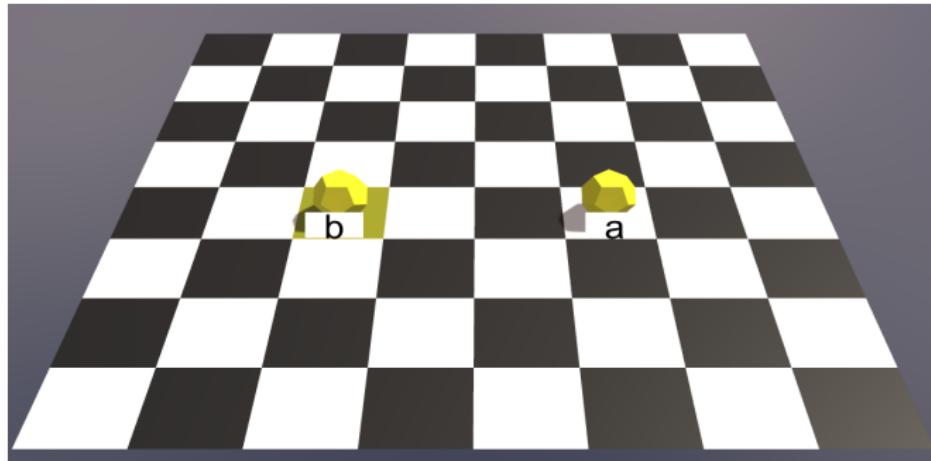
Wahrheitswertbelegungen und Bivalence World

- Komplexe Sätze können auch zur Beschreibung von Welten in Bivalence World verwendet werden
- Jede Bivalence World-Welt führt zu einer Belegung der atomaren Sätze mit Wahrheitswerten
 - d.h. zu einer Wahrheitswertbelegung

Definition

Eine Wahrheitswertbelegung h über einer Menge \mathcal{A} von atomaren Sätzen aus der Sprache von Bivalenz World heißt **BW-kompatibel**, wenn es eine Bivalenz World-Welt gibt, die die gleichen Sätze aus \mathcal{A} mit “wahr” bewertet wie h .

Eine BW-kompatible Wahrheitswertbelegung



$$h(Cube(a)) = F \quad h(Large(a)) = F \quad h(LeftOf(a, b)) = F$$

$$\hat{h}(Cube(a)) = F \quad \hat{h}(Large(a)) = F \quad \hat{h}(LeftOf(a, b)) = F$$

$$\hat{h}(Cube(a) \vee Large(a)) = F$$

$$\hat{h}((Cube(a) \vee Large(a)) \rightarrow LeftOf(a, b)) = T$$

Wahrheitstafel für Bivalence World: Beispiel

$h(\text{Cube}(a))$	$h(\text{Tet}(a))$	$\hat{h}(\text{Cube}(a) \wedge \text{Tet}(a))$	$\hat{h}(\neg(\text{Cube}(a) \wedge \text{Tet}(a)))$
T	T	T	F
T	F	F	T
F	T	F	T
F	F	F	T

rot: nicht BW-kompatibel

Wahrheit und Erfüllbarkeit in Bivalence World

Bivalence World-Wahrheiten: Motivation

- Logisch notwendige Sätze oder logische Wahrheiten sind Sätze, die unter allen Umständen (in allen Welten) wahr sind.
- Logische Wahrheiten wie $A \vee \neg A$ sind aus rein logischen Gründen wahr.
 - Damit sind sie eher langweilig, denn sie sagen nichts über die Welt aus, sondern nur etwas über die Logik selbst.
- Bivalence World-Wahrheiten wie $\text{Cube}(a) \vee \text{Dodec}(a) \vee \text{Tet}(a)$ sagen etwas über die Domäne Bivalence World aus.

Bivalence World-Wahrheiten: Definition

Definition

Ein Satz P ist

- ① Wahrheitstafel-wahr, WT-wahr oder eine **Tautologie**, wenn er wahr ist für alle Wahrheitswertbelegungen, d.h. für alle Wahrheitswertbelegungen h gilt $\hat{h}(P) = T$
- ② **Bivalence World-wahr** oder **BW-wahr**, wenn er in allen Welten von Bivalence World wahr ist.

Der Unterschied liegt in der **Menge der betrachteten Welten**:

- ① alle **Wahrheitswertbelegungen** = **Zeilen** der Wahrheitstafel
- ② alle **Klötzchenwelten** von **Bivalence World**
(äquivalent: nur **BW-kompatible Wahrheitswertbelegungen**)

Wahrheit in Bivalence World: Beispiel

$h(\text{Cube}(a))$	$h(\text{Tet}(a))$	$\hat{h}(\text{Cube}(a) \wedge \text{Tet}(a))$	$\hat{h}(\neg(\text{Cube}(a) \wedge \text{Tet}(a)))$
T	T	T	F
T	F	F	T
F	T	F	T
F	F	F	T

rot: nicht BW-kompatibel

- $\neg(\text{Cube}(a) \wedge \text{Tet}(a))$ ist unter allen BW-kompatiblen Wahrheitswertbelegungen wahr, also eine **BW-Wahrheit**.
- $\neg(\text{Cube}(a) \wedge \text{Tet}(a))$ ist nicht unter der Wahrheitswertbelegung wahr, die der **ersten Zeile** entspricht, also **keine Tautologie**.
 - Diese Wahrheitswertbelegung ist nicht BW-kompatibel, aber in der reinen Aussagenlogik dennoch zulässig.

Logische Wahrheiten

Theorem

Jede Tautologie ist eine BW-Wahrheit.

Beweis: jede Tautologie ist für alle Wahrheitswertbelegungen wahr, also erst recht für alle aus Bivalence World resultierenden (d.h. BW-kompatiblen) Wahrheitswertbelegungen.

Es gibt umgekehrt BW-Wahrheiten, die keine Tautologien sind.

Logische Wahrheiten: Beispiele

Es gibt drei Möglichkeiten. Ein Satz ist

- ① eine Tautologie, und damit nach vorigem Theorem auch BW-wahr
- ② BW-wahr, aber keine Tautologie
- ③ weder BW-wahr noch Tautologie

Beispiele:

- $A \rightarrow A \rightsquigarrow 1$
- $\text{Cube}(a) \rightarrow \text{Large}(a) \rightsquigarrow 3$
- $\text{Cube}(a) \rightarrow \neg \text{Dodec}(a) \rightsquigarrow 2$
- $\text{Cube}(a) \vee \neg \text{Cube}(a) \rightsquigarrow 1$
- $\text{Small}(a) \vee \text{Medium}(a) \vee \text{Large}(a) \rightsquigarrow 2$
- $(\text{Large}(a) \wedge \text{SameSize}(a, b)) \rightarrow \text{Large}(b) \rightsquigarrow 2$
- $\perp \rightsquigarrow 3$

Erfüllbarkeit: Definition

Definition

Eine logische Theorie \mathcal{T} ist

- ① **Wahrheitstafel-erfüllbar** (WT-erfüllbar) wenn für mindestens eine Wahrheitswertbelegung alle Sätze wahr sind, d.h. es gibt eine Wahrheitswertbelegung h mit $\hat{h}(P)$ für alle $P \in \mathcal{T}$.
- ② **Bivalence World-erfüllbar** oder **BW-erfüllbar**, wenn in mindestestens einer Bivalence World-Welt alle Sätze aus \mathcal{T} wahr sind.

Der Unterschied liegt wiederum in der **Menge der betrachteten Welten**:

- ① alle **Wahrheitswertbelegungen** = **Zeilen** der Wahrheitstafel
- ② alle **Klötzchenwelten von Bivalence World**
(äquivalent: nur BW-kompatible Wahrheitswertbelegungen)

Verschiedene Formen von Erfüllbarkeit

Theorem

Jede BW-erfüllbare logische Theorie ist auch WT-erfüllbar.

Beweis: eine BW-erfüllbare logische Theorie ist in einer aus Bivalence World resultierenden (d.h. BW-kompatiblen) Wahrheitswertbelegung wahr. Also ist sie auch für eine Wahrheitswertbelegung wahr, also WT-erfüllbar.

Es gibt umgekehrt WT-erfüllbare logischen Theorien, die nicht BW-erfüllbar sind.

Erfüllbarkeit: Beispiele

Es gibt drei Möglichkeiten. Eine logische Theorie ist

- ① BW-erfüllbar, und damit nach vorigem Theorem auch WT-erfüllbar
- ② WT-erfüllbar, aber nicht BW-erfüllbar
- ③ weder BW-erfüllbar noch WT-erfüllbar

Beispiele:

- $\{A, \neg A\} \rightsquigarrow 3$
- $\{\text{Cube}(a) \rightarrow \text{Large}(a)\} \rightsquigarrow 1$
- $\{\text{Cube}(a), \text{Dodec}(a)\} \rightsquigarrow 2$
- $\{\text{Cube}(a) \vee \text{Dodec}(a), \neg \text{Cube}(a), \neg \text{Dodec}(a)\} \rightsquigarrow 3$
- $\{\text{Cube}(a) \rightarrow \neg \text{Cube}(a)\} \rightsquigarrow 1$
- $\{\text{Large}(a), \text{Medium}(a)\} \rightsquigarrow 2$
- $\{\perp\} \rightsquigarrow 3$

Zusammenfassung 3

Zusammenfassung Veranstaltung 3 und Ausblick

Was haben wir heute gelernt?

- Semantik aussagenlogischer Sätze
- Logische Wahrheit und Erfüllbarkeit auf verschiedenen Ebenen
- Die Wahrheitstafelmethode

Nächstes Mal behandeln wir:

- Logische Folgerungen
- Logische Äquivalenzen

Lernziele Veranstaltung 4

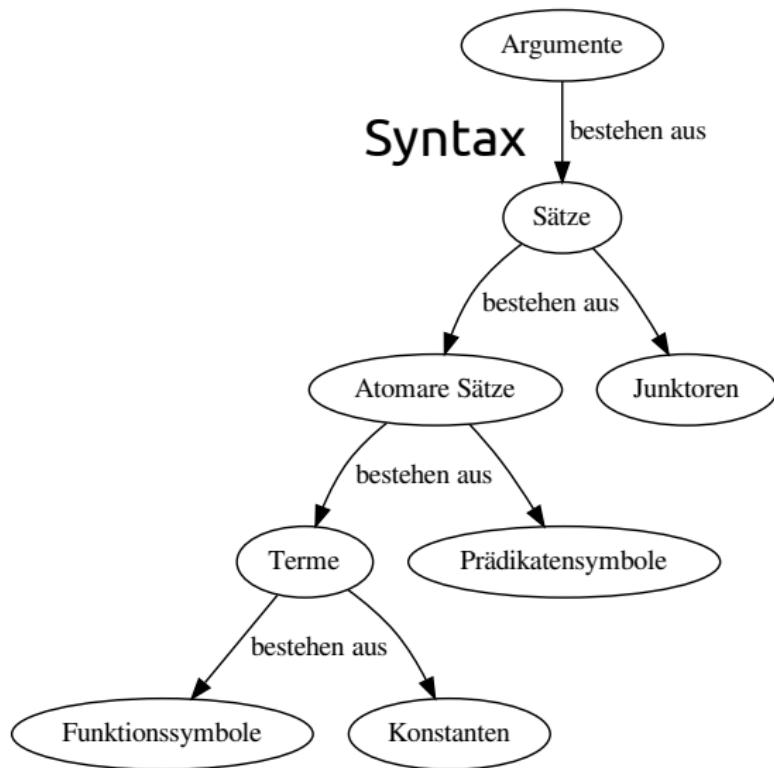
Lernziele für heute (Veranstaltung 4)

- ich vertiefe mein Verständnis der **Semantik** aussagenlogischer Sätze
- ich verstehe den Begriff der **logischen Folgerung**
- ich kann Gültigkeit von einfachen Argumenten **zeigen** und **widerlegen**
- ich verstehe den Begriff der **logischen Äquivalenz**
- ich kenne **grundlegende Äquivalenzen** der Aussagenlogik

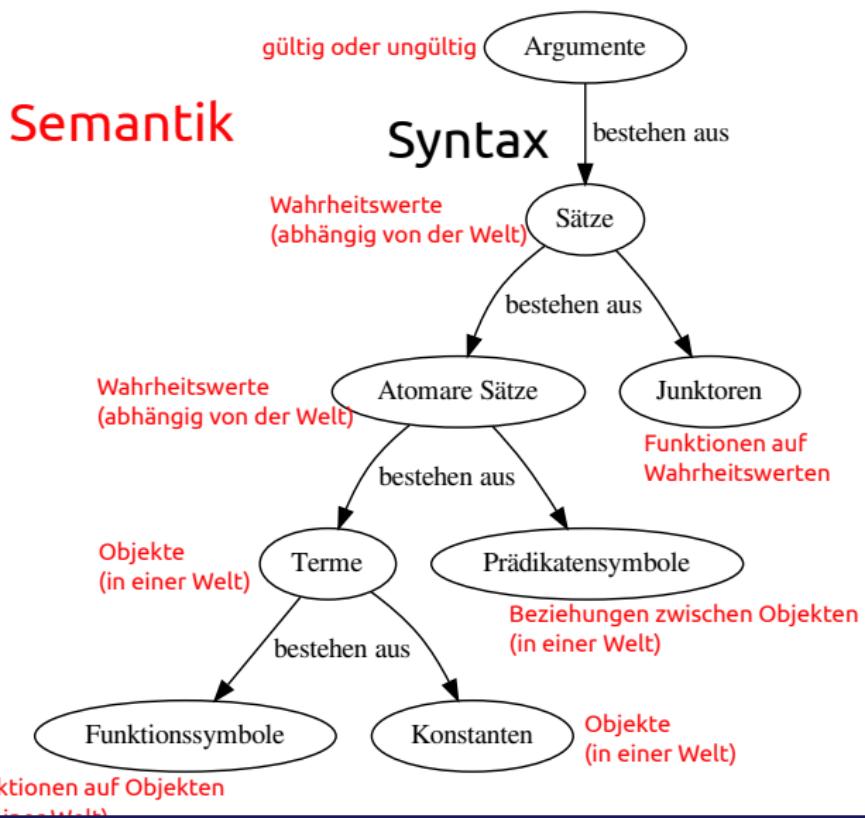
Buch: 2.5, 3.6, 4.2, 4.3, 4.5

Logische Folgerung

Syntax von Sätzen und Argumenten



Syntax und Semantik von Sätzen und Argumenten



Logische Folgerungen: Motivation

Ein logisches Argument ist gültig (oder eine logische Folgerung), wenn unter allen Umständen, unter denen die Prämissen wahr sind, auch die Konklusion wahr ist. (Folie 23)

Mit Hilfe der Syntax und Semantik der Aussagenlogik können wir diesen Begriff nun präzisieren.

Tautologische Folgerung: Definition

Definition

Ein Satz Q (Konklusion) ist eine **tautologische Folgerung** aus Prämissen P_1, \dots, P_n , wenn für alle Wahrheitsbelegungen der atomaren Sätze, die P_1, \dots, P_n wahr machen, auch Q wahr ist, d.h. für alle Wahrheitsbelegungen h mit $\hat{h}(P_1) = \dots = \hat{h}(P_n) = T$ gilt auch $\hat{h}(Q) = T$.

- Test mit Wahrheitstafel (Programm Boole):
In jeder Zeile, in der **alle Prämissen wahr** sind, muss auch die **Konklusion wahr** sein.
- anderer, äquivalenter Test mit Wahrheitstafel:
In jeder Zeile, in der die **Konklusion falsch** ist, muss auch **mindestens eine Prämisze falsch** sein.
- Wenn diese Tests fehlschlagen, gilt es ein **Gegenbeispiel**, d.h. eine Zeile der Wahrheitstafel (=Wahrheitswertbelegung), die die **Prämissen wahr** macht und die **Konklusion falsch**.

Welches sind tautologische Folgerungen?

- ① | Larger(a, b)
 | Larger(b, c)
 | Larger(a, c)

- ② | $a = b$
 | $b = c$
 | $a = c$

- ③ | $\text{Cube}(a) \vee \text{Tet}(a)$
 | $\neg \text{Cube}(a)$
 | $\text{Tet}(a)$

BW-Logische Folgerung

Definition

Ein Satz Q (Konklusion) ist eine **BW-logische Folgerung** aus Prämissen P_1, \dots, P_n , wenn in allen Welten von Bivalence World, in denen P_1, \dots, P_n wahr sind, auch Q wahr ist.

Test auf BW-logische Folgerung:

- **Argumentiere informell**, warum wahre Prämissen auch zu einer wahren Konklusion führen (später werden wir **formale Beweismethoden** kennenlernen).
- oder finde ein **Gegenbeispiel**, d.h. eine Bivalence World-Welt, die die Prämissen wahr macht, die Konklusion aber falsch.

Tautologische versus BW-Folgerung

Der Unterschied von tautologischen Folgerung und BW-Folgerung liegt wiederum in der **Menge der betrachteten Welten**:

tautologische Folgerung: alle **Wahrheitswertbelegungen**
= **Zeilen** der Wahrheitstafel

BW-Folgerung: alle **Klötzchenwelten** von Bivalence World
(äquivalent: nur BW-kompatible
Wahrheitswertbelegungen)

Theorem

*Wenn Q eine tautologische Folgerung aus P_1, \dots, P_n ist,
dann ist Q auch eine **BW-logische Folgerung** aus P_1, \dots, P_n .*

Es gibt umgekehrt BW-logische Folgerungen, die keine tautologischen Folgerungen sind.

Tautologische versus BW-Folgerung

Frage: welche der folgenden Argumente sind tautologische Folgerungen, welche BW-logische Folgerungen?

- ①
 - | Larger(a, b)
 - | Larger(b, c)
 - | Larger(a, c)

- ②
 - | a = b
 - | b = c
 - | a = c

- ③
 - | Cube(a) ∨ Tet(a)
 - | \neg Cube(a)
 - | Tet(a)

Sind die folgenden Argumente gültig (in Bivalence World)?

Small(a)
Larger(b, a)
Large(b)

Small(a)
Larger(a, b)
Large(b)

Logische Folgerung und Konditionale

Theorem (“Deduktionstheorem”)

Ein Satz Q ist eine tautologische Folgerung aus P_1, \dots, P_n , genau dann wenn

$$(P_1 \wedge \cdots \wedge P_n) \rightarrow Q$$

eine Tautologie ist.

Ein Satz Q ist eine BW-logische Folgerung aus P_1, \dots, P_n , genau dann wenn

$$(P_1 \wedge \cdots \wedge P_n) \rightarrow Q$$

eine BW-Wahrheit ist.

Bemerkung:

- → ist ein Junktor, mit dem man Sätze aufbauen kann.
- “logische Folgerung” ist kein Satz, sondern eine Meta-Aussage über eine Beziehung zwischen Sätzen.

Deduktionstheorem: Beispiel

|
| Larger(a, b)
| Larger(b, c)
| Larger(a, c)

ist eine tautologische Folgerung (bzw. BW-Folgerung) genau dann,
wenn

$$(Larger(a, b) \wedge Larger(b, c)) \rightarrow Larger(a, c)$$

eine Tautologie (bzw. BW-Wahrheit) ist.

Logische Folgerung und logische Wahrheit

Theorem

Ein Satz P ist eine Tautologie genau dann wenn er aus der leeren Prämissenmenge tautologisch folgt.

Ein Satz P ist eine BW-Wahrheit genau dann wenn er eine BW-Folgerung aus der leeren Prämissenmenge ist.

Logische Folgerung und logische Wahrheit: Beispiel

$$(Larger(a, b) \wedge Larger(b, c)) \rightarrow Larger(a, c)$$

ist eine Tautologie (bzw. BW-Wahrheit) genau dann, wenn



$$(Larger(a, b) \wedge Larger(b, c)) \rightarrow Larger(a, c)$$

eine tautologische Folgerung (bzw. BW-Folgerung) ist.

Logische Folgerung und Erfüllbarkeit

Theorem

Eine logische Theorie $\{P_1, \dots, P_n\}$ ist **WT-erfüllbar** genau dann wenn \perp nicht aus P_1, \dots, P_n tautologisch folgt.

Eine logische Theorie $\{P_1, \dots, P_n\}$ ist **BW-erfüllbar** genau dann wenn \perp keine BW-Folgerung aus P_1, \dots, P_n ist.

Logische Äquivalenzen

Logische Äquivalenz: Motivation

- Zwei Sätze sind **logisch äquivalent**, wenn sie in allen Welten den gleichen Wahrheitswert haben.
- Logische Äquivalenzen ermöglichen uns, Sätze umzuformen
 - damit können z.B. **Normalformen** realisiert werden (siehe nächste Woche).

Logische Äquivalenz: Definition

Definition

Zwei Sätze P und Q heißen **tautologisch äquivalent** (in Zeichen $P \Leftrightarrow Q$), wenn sie für alle Wahrheitswertbelegungen der atomaren Sätze den gleichen Wahrheitswert haben, d.h. für alle Wahrheitswertbelegungen h gilt: $\hat{h}(P) = \hat{h}(Q)$.

Um tautologische Äquivalenz zu prüfen, teste, ob P und Q in jeder Zeile der **Wahrheitstafel** den gleichen Wert erhalten. Dies geht z.B. mit Boole.

Bekannte tautologische Äquivalenzen (Boolesche Algebra)

$\neg\neg P$	\Leftrightarrow	P	doppelte Negation
$(P \vee P)$	\Leftrightarrow	P	Idempotenz von \vee
$(P \wedge P)$	\Leftrightarrow	P	Idempotenz von \wedge
$(P \vee Q)$	\Leftrightarrow	$(Q \vee P)$	Kommutativität von \vee
$(P \wedge Q)$	\Leftrightarrow	$(Q \wedge P)$	Kommutativität von \wedge
$(P \vee Q) \vee R$	\Leftrightarrow	$P \vee (Q \vee R)$	Assoziativität von \vee
$(P \wedge Q) \wedge R$	\Leftrightarrow	$P \wedge (Q \wedge R)$	Assoziativität von \wedge
$(P \vee (P \wedge Q))$	\Leftrightarrow	P	Absorption 1
$(P \wedge (P \vee Q))$	\Leftrightarrow	P	Absorption 2
$P \wedge (Q \vee R)$	\Leftrightarrow	$(P \wedge Q) \vee (P \wedge R)$	Distributivität 1
$P \vee (Q \wedge R)$	\Leftrightarrow	$(P \vee Q) \wedge (P \vee R)$	Distributivität 2
$\neg(P \wedge Q)$	\Leftrightarrow	$(\neg P \vee \neg Q)$	de Morgan 1
$\neg(P \vee Q)$	\Leftrightarrow	$(\neg P \wedge \neg Q)$	de Morgan 2
$P \vee \neg P$	\Leftrightarrow	$\neg \perp$	Ausgeschlossenes Drittels

Kompakte Schreibweise

- Kompakte Schreibweise:

- Die äußereren Klammern können weggelassen werden.
- Zusätzliche Klammern dürfen eingeführt werden.
- Bei mehrfachen Disjunktionen oder Konjunktionen können die Klammern weggelassen werden: $P \vee Q \vee R, P \wedge Q \wedge R$.

Der letzte Punkt ist darin begründet, dass Disjunktionen und Konjunktionen assoziativ sind
(siehe Äquivalenzen auf der vorigen Folie).

- $(\text{Cube}(a) \vee (\text{Tet}(a) \vee \text{Dodec}(a)))$

kann abgekürzt werden als:

$$\text{Cube}(a) \vee \text{Tet}(a) \vee \text{Dodec}(a)$$

- Auf diese Weise können Disjunktion und Konjunktion auch als mehrstellige Junktoren aufgefasst werden (wichtig für Normalformen; siehe die nächste Veranstaltung).

Tautologische Äquivalenzen von (Bi)Konditional und Falsum

$$P \rightarrow Q \Leftrightarrow \neg Q \rightarrow \neg P \quad \text{Kontraposition}$$

$$P \rightarrow Q \Leftrightarrow \neg P \vee Q \quad \text{Def. } \rightarrow$$

$$\neg(P \rightarrow Q) \Leftrightarrow P \wedge \neg Q$$

$$P \leftrightarrow Q \Leftrightarrow (P \rightarrow Q) \wedge (Q \rightarrow P) \quad \text{Def. } \leftrightarrow$$

$$P \leftrightarrow Q \Leftrightarrow (\neg P \vee Q) \wedge (P \vee \neg Q)$$

$$P \leftrightarrow Q \Leftrightarrow (P \wedge Q) \vee (\neg P \wedge \neg Q)$$

$$\perp \Leftrightarrow P \wedge \neg P \quad \text{Def. } \perp$$

$$P \vee \perp \Leftrightarrow P \quad P \wedge \top \Leftrightarrow P$$

$$P \rightarrow \perp \Leftrightarrow \neg P \quad \perp \rightarrow P \Leftrightarrow \top$$

$$P \rightarrow \top \Leftrightarrow \top \quad \top \rightarrow P \Leftrightarrow P$$

Logische Äquivalenz in Bivalence World

Definition

Zwei Sätze P und Q heißen

- ① **tautologisch äquivalent** (in Zeichen $P \Leftrightarrow Q$), wenn sie für alle Wahrheitswertbelegungen der atomaren Sätze den gleichen Wahrheitswert haben, d.h. für alle Wahrheitswertbelegungen h gilt: $\hat{h}(P) = \hat{h}(Q)$.
- ② **BW-äquivalent** (in Zeichen $P \Leftrightarrow_{BW} Q$), wenn sie in allen Welten von Bivalence World den gleichen Wahrheitswert haben.

Der Unterschied liegt wiederum in der **Menge der betrachteten Welten**:

- ① alle **Wahrheitswertbelegungen** = **Zeilen** der Wahrheitstafel
- ② alle **Klötzchenwelten** von Bivalence World
(äquivalent: nur BW-kompatible Wahrheitswertbelegungen)

Bivalence World-Äquivalenzen

$$\neg \text{Cube}(a) \wedge \neg \text{Tet}(a) \Leftrightarrow_{BW} \text{Dodec}(a)$$

$$\neg \text{Small}(a) \wedge \neg \text{Medium}(a) \Leftrightarrow_{BW} \text{Large}(a)$$

$$\text{Larger}(a, b) \Leftrightarrow_{BW} \text{Smaller}(b, a)$$

$$\text{LeftOf}(a, b) \Leftrightarrow_{BW} \text{RightOf}(b, a)$$

Dies sind alles keine tautologischen Äquivalenzen.

Theorem

Jede tautologische Äquivalenz ist auch eine BW-Äquivalenz.

Äquivalenz versus Bikonditional

Theorem

P und Q sind tautologisch äquivalent ($P \Leftrightarrow Q$)

genau dann, wenn

der Satz $P \leftrightarrow Q$ eine Tautologie ist.

Theorem

P und Q sind BW-äquivalent

genau dann, wenn

der Satz $P \leftrightarrow Q$ eine BW-Wahrheit ist.

Bemerkung:

- $P \Leftrightarrow Q$ ist kein Satz in PL1, sondern eine Meta-Aussage.
- “ \leftrightarrow ” ist Junktor, Operation in der Menge der PL1-Sätze,
“ \Leftrightarrow ” ist eine Relation in der Menge der PL1-Sätze.

Äquivalenz versus Bikonditional: Beispiel

Es gilt

$$\neg \text{Cube}(a) \wedge \neg \text{Tet}(a) \Leftrightarrow_{BW} \text{Dodec}(a)$$

und daher ist

$$(\neg \text{Cube}(a) \wedge \neg \text{Tet}(a)) \leftrightarrow_{BW} \text{Dodec}(a)$$

eine BW-Wahrheit.

Zusammenfassung Veranstaltung 4

Zusammenfassung Veranstaltung 4 und Ausblick

Was haben wir heute gelernt?

- Semantik aussagenlogischer Sätze mittels Wahrheitstafeln
- Logische Wahrheit und Folgerung auf verschiedenen Ebenen
- Die Wahrheitstafelmethode
- Logische Äquivalenzen

Nächstes Mal behandeln wir:

- Normalformen (basierend auf den heute kennengelernten tautologischen Äquivalenzen)
- wie viele Junktoren brauchen wir?

Lernziele Veranstaltung 5

Lernziele für heute (Veranstaltung 5)

- ich kann Sätze in verschiedene **Normalformen** umwandeln
 - Sinn: viele Beweiser und SAT-Solver verwenden Normalformen
 - ich weiß, mit **wie vielen Junktoren** man auskommt

Buch: 4.5, 4.6, 7.4

Normalformen

Motivation für Normalformen

Viele Werkzeuge verlangen Eingaben in **konjunktiver Normalform**:

- **SAT-Solver** sind ein effizientes Werkzeug für Testen von Erfüllbarkeit und logischer Folgerung in der Aussagenlogik. Mehr dazu nächste Woche.
- **Resolutionskalküle** sind Beweiskalküle, die effizient als Software implementierbar sind.
- Werkzeuge zum **Schaltkreisentwurf**.
- **Prolog** (Programmieren in Logik) verwendet Klauseln, die auf konjunktiver Normalform basieren.

Wiederholung: tautologische Äquivalenzen

$$P \rightarrow Q \Leftrightarrow \neg P \vee Q \quad \text{Def. } \rightarrow$$

$$P \leftrightarrow Q \Leftrightarrow (P \rightarrow Q) \wedge (Q \rightarrow P) \quad \text{Def. } \leftrightarrow$$

$$\perp \Leftrightarrow P \wedge \neg P \quad \text{Def. } \perp$$

$$\neg(P \wedge Q) \Leftrightarrow (\neg P \vee \neg Q) \quad \text{de Morgan 1}$$

$$\neg(P \vee Q) \Leftrightarrow (\neg P \wedge \neg Q) \quad \text{de Morgan 2}$$

$$\neg\neg P \Leftrightarrow P \quad \text{doppelte Negation}$$

$$P \wedge (Q \vee R) \Leftrightarrow (P \wedge Q) \vee (P \wedge R) \quad \text{Distributivität 1}$$

$$P \vee (Q \wedge R) \Leftrightarrow (P \vee Q) \wedge (P \vee R) \quad \text{Distributivität 2}$$

Substitution von Äquivalentem

Substitution von Äquivalentem: Wenn P und Q tautologisch äquivalent sind ($P \Leftrightarrow Q$), dann sind die Sätze, die sich ergeben, wenn das eine im Kontext eines größeren Satzes für das andere ersetzt wird, auch tautologisch äquivalent, also $S \Leftrightarrow S[P \mapsto Q]$.

Wir können also de Morgan und das Gesetz der doppelten Negation auch im Kontext größerer Sätze anwenden.

Negations-Normalform (NNF)

Definition

Ein Satz ist in Negations-Normalform (NNF), falls er nur die Junktoren $\{\vee, \wedge, \neg\}$ benutzt und sich alle Vorkommen von \neg direkt auf atomare Sätze beziehen.

Theorem

Für jeden aussagenlogischen Satz gibt es einen tautologisch äquivalenten Satz in NNF.

Beweisidee: zunächst können $\{\rightarrow, \leftrightarrow, \perp\}$ mittels Def. \rightarrow , Def. \leftrightarrow und Def. \perp eliminiert werden. Über schrittweise Anwendung der de Morgan-Äquivalenzen lassen sich dann die Negationszeichen ganz nach innen schieben. Doppelte Negationen können eliminiert werden.

Negations-Normalform: Beispiel

Was ist die NNF von $\neg(\neg(A \vee B) \wedge (C \vee D))$?

Konjunktive Normalform: Definition

Definition

Ein **Literal** ist ein atomarer Satz oder die Negation eines atomaren Satzes.

Definition

Ein Satz ist in **konjunktiver Normalform** (KNF), wenn er eine Konjunktion aus einer oder mehreren Disjunktionen von einem oder mehreren Literalen ist. Konjunktionen und Disjunktionen können dabei auch eingliedrig sein; der Junktor wird dann weggelassen.

Beispiele in KNF:

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

$$A \wedge (\neg C \vee \neg D)$$

$$A \vee \neg B$$

$$A \wedge \neg C$$

$$A$$

Beispiele nicht in KNF:

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

$$\neg(A \wedge (\neg C \vee \neg D))$$

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

$$A \rightarrow C$$

$$\neg\neg A$$

Konjunktive Normalform: Theorem

Theorem

Für jeden aussagenlogischen Satz gibt es einen tautologisch äquivalenten Satz in KNF.

Beweisidee: Durch schrittweise Anwendung der Distributivität von \vee über \wedge lässt sich ein beliebiger Satz in Negations-Normalform in eine konjunktive Normalform überführen.

Konjunktive Normalform: Beispiel

Was ist die KNF von $\neg((A \rightarrow B) \wedge C)$?

Disjunktive Normalform: Definition

Definition

Ein Satz ist in **disjunktiver Normalform** (DNF), wenn er eine Disjunktion aus einer oder mehreren Konjunktionen von einem oder mehreren Literalen ist. Konjunktionen und Disjunktionen können dabei auch eingliedrig sein; sie werden dann weggelassen.

Beispielsätze in DNF:

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

$$A \vee (\neg C \wedge \neg D)$$

$$A \vee \neg B$$

$$A \wedge \neg C$$

$$A$$

Beispielsätze nicht in DNF:

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

$$\neg(A \wedge (\neg C \vee \neg D))$$

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

$$A \rightarrow C$$

$$\neg\neg A$$

Disjunktive Normalform: Theorem

Theorem

Für jeden aussagenlogischen Satz gibt es einen tautologisch äquivalenten Satz in DNF.

Beweisidee: Durch schrittweise Anwendung der Distributivität von \wedge über \vee lässt sich ein beliebiger Satz in Negations-Normalform in eine disjunktive Normalform überführen.

Disjunktive Normalform: Beispiel

Was ist die DNF von $\neg(\neg(A \vee B) \vee (C \wedge D))$?

Wahrheitsfunktionale Vollständigkeit

Wahrheitsfunktionale Vollständigkeit: Motivation

- Bisher haben wir das einschließende Oder verwendet
 - mit T oder $T = T$.

Häufig verwendet man im Deutschen ein ausschließendes Oder

- mit T xor $T = F$.

- Dieses ausschließende Oder ließe sich zu unserer Logik als neuer Junktor hinzufügen, dessen Semantik mit Hilfe einer Wahrheitstabelle definiert wird. Oder könne wir es mittels der vorhandenen Junktoren ausdrücken?
 - Der Begriff der **wahrheitsfunktionalen Vollständigkeit** liefert die Antwort auf diese Frage.

Wahrheitsfunktionale Junktoren

Definition

Ein logischer Junktor heißt **wahrheitsfunktional**, wenn der Wahrheitswert eines komplexen Satzes, der mittels dieses Junktors gebildet wird, nur von den Wahrheitswerten der einfacheren Sätze, aus denen er aufgebaut ist, abhängt.

Wahrheitsfunktionale Junktoren: \wedge , \vee , \neg , \leftarrow , \leftrightarrow

Nicht wahrheitsfunktional: da, nachdem, notwendigerweise

Wahrheitsfunktionale Vollständigkeit

Definition

Eine Menge von Junktoren heißt **wahrheitsfunktional vollständig**, wenn sich mit ihr jede Wahrheitsfunktion ausdrücken lässt. D.h. mit ihnen lässt sich jeder beliebige wahrheitsfunktionale Junktator darstellen.

Theorem

Die Menge $\{\wedge, \vee, \neg\}$ ist wahrheitsfunktional vollständig.

Theorem

Die Menge $\{\vee, \neg\}$ ist wahrheitsfunktional vollständig.

Theorem

Die Menge $\{\wedge, \neg\}$ ist wahrheitsfunktional vollständig.

Beispiel: ein zweistelliger wahrheitsfunktionaler Junktor

P	Q	Weder P noch Q
T	T	F
T	F	F
F	T	F
F	F	T

$$\neg P \wedge \neg Q$$

“Weder P noch Q ” kann durch $\neg P \wedge \neg Q$ ausgedrückt werden.

Beispiel: ein dreistelliger wahrheitsfunktionaler Junktor

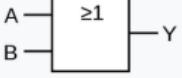
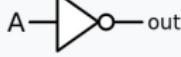
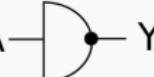
P	Q	R	$\clubsuit(P, Q, R)$	
T	T	T	T	$P \wedge Q \wedge R$
T	T	F	T	$P \wedge Q \wedge \neg R$
T	F	T	F	
T	F	F	F	
F	T	T	T	$\neg P \wedge Q \wedge R$
F	T	F	F	
F	F	T	T	$\neg P \wedge \neg Q \wedge R$
F	F	F	F	

$\clubsuit(P, Q, R)$ kann ausgedrückt werden durch

$$(P \wedge Q \wedge R) \vee (P \wedge Q \wedge \neg R) \vee (\neg P \wedge Q \wedge R) \vee (\neg P \wedge \neg Q \wedge R).$$

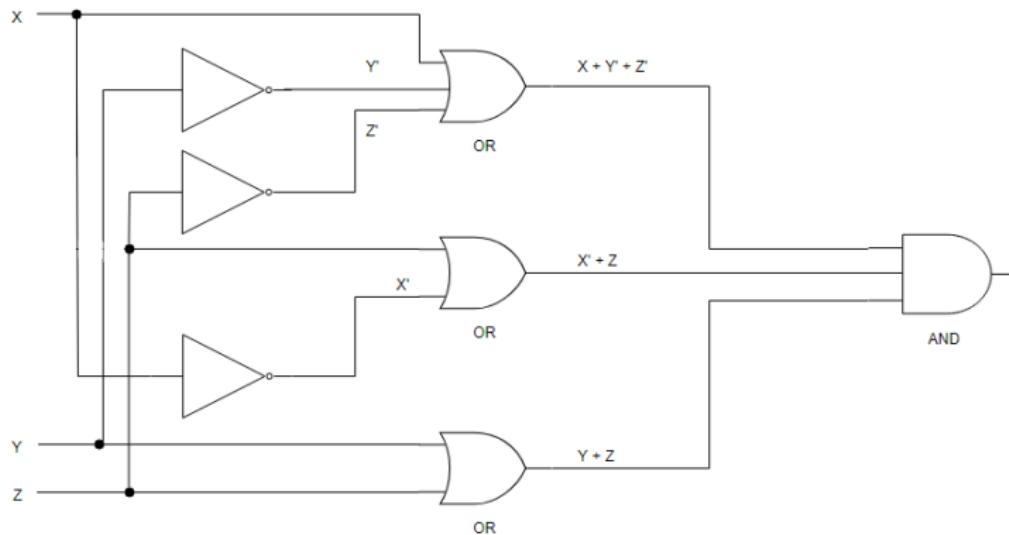
Exkurs: Logik-Gatter

Gatter als weitere Syntax für Aussagenlogik

		IEC 60617-12 : 1997 & ANSI/IEEE Std 91/91a-1991	ANSI/IEEE Std 91/91a-1991	DIN 40700 (vor 1976)																
Und-Gatter (AND)	$Y = A \wedge B$ $Y = A \cdot B$ $Y = AB$ $Y = A \& B$				<table border="1" data-bbox="1125 322 1226 530"> <tr><th>A</th><th>B</th><th>Y</th></tr> <tr><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>1</td><td>0</td></tr> <tr><td>1</td><td>0</td><td>0</td></tr> <tr><td>1</td><td>1</td><td>1</td></tr> </table>	A	B	Y	0	0	0	0	1	0	1	0	0	1	1	1
A	B	Y																		
0	0	0																		
0	1	0																		
1	0	0																		
1	1	1																		
Oder-Gatter (OR)	$Y = A \vee B$ $Y = A + B$				<table border="1" data-bbox="1125 578 1226 786"> <tr><th>A</th><th>B</th><th>Y</th></tr> <tr><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>1</td><td>1</td></tr> <tr><td>1</td><td>0</td><td>1</td></tr> <tr><td>1</td><td>1</td><td>1</td></tr> </table>	A	B	Y	0	0	0	0	1	1	1	0	1	1	1	1
A	B	Y																		
0	0	0																		
0	1	1																		
1	0	1																		
1	1	1																		
Nicht-Gatter (NOT)	$Y = \bar{A}$ $Y = \neg A$ $Y = \tilde{A}$				<table border="1" data-bbox="1125 834 1226 960"> <tr><th>A</th><th>Y</th></tr> <tr><td>0</td><td>1</td></tr> <tr><td>1</td><td>0</td></tr> </table>	A	Y	0	1	1	0									
A	Y																			
0	1																			
1	0																			



Schaltkreise $\hat{=}$ logische Sätze

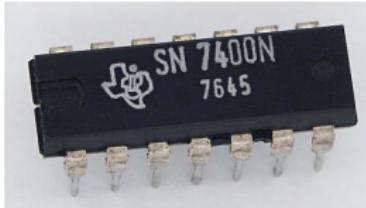
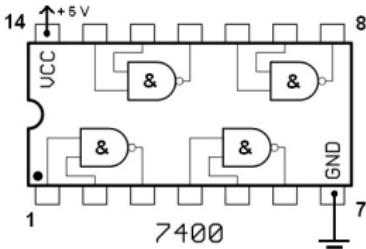


$$(X \vee \neg Y \vee \neg Z) \wedge (\neg X \vee Z) \wedge (Y \vee Z)$$

Anders als bei Sätzen können bei Gattern Teilausdrücke effizient mehrfach verwendet werden.

Gatter und wahrheitsfunktionale Vollständigkeit

- wahrheitsfunktionale Vollständigkeit sagt uns, dass Und-, Oder- und Nicht-Gatter ausreichen, um alle logischen Schaltungen zu realisieren
- es reichen bereits NAND-Gatter aus
- es reichen bereits NOR-Gatter aus



Zusammenfassung Veranstaltung 5

Zusammenfassung Veranstaltung 5 und Ausblick

Was haben wir heute gelernt?

- konjunktive, disjunktive und Negations-**Normalform**
- $\{\vee, \neg\}$ und $\{\wedge, \neg\}$ sind **wahrheitsfunktional vollständig**.
- **Gatter** als alternative Syntax für aussagenlogische Sätze.

Nächstes Mal behandeln wir:

- Beweise im Fitch-Kalkül

Lernziele Veranstaltung 6

Lernziele für heute (Veranstaltung 6)

- ich kann mit dem **Fitch-Kalkül** und dem Programm Fitch umgehen
- ich lerne, **Beweise** vollständig präzise aufzuschreiben
- ich lerne die Funktionsweise eines **formalen Regelsysteme** kennen (diese werden in der Informatik häufig benutzt)

Buch: 2.2–2.4, 5.1–5.3, 6.1–6.4

Beweismethoden

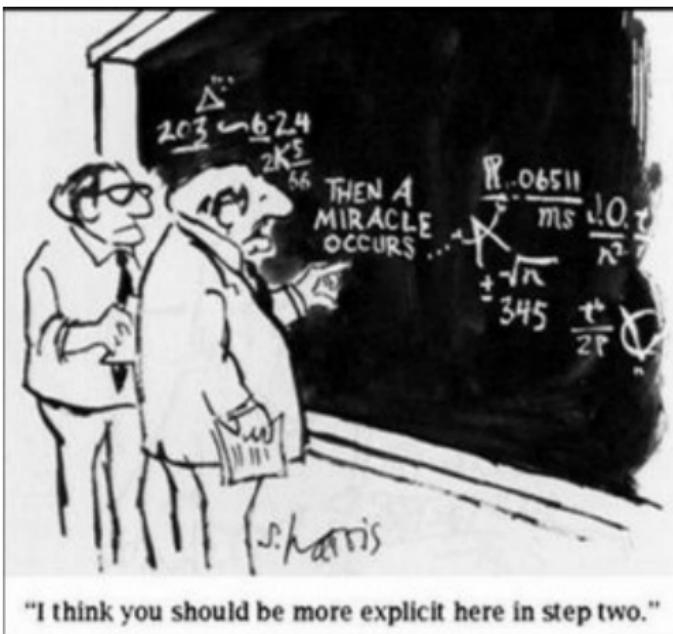
Grenzen der Wahrheitstafelmethode

- ① die Wahrheitstafelmethode führt zu **exponentiell wachsenden Tabellen**.
 - 20 atomare Formeln \Rightarrow mehr als 1 000 000 Zeilen.
- ② die Wahrheitstafelmethode kann nicht für die **Logik erster Stufe** erweitert werden.
 - **Model Checking** kann die erste Beschränkung überwinden (bis zu 1 000 000 atomaren Formeln).
 - **Beweise** können beide Beschränkungen überwinden.

Beweise

- Ein Beweis besteht aus einer Folge von **Beweisschritten**.
- Jeder Beweisschritt muss gültig sein und muss in
 - **informellen** Beweisen bedeutsam aber leicht zu verstehen sein,
 - **formalen** Beweisen durch **Beweisregeln** gezeigt werden.
- Folgende gültige Schlussprinzipien dürfen in informellen (aber nicht formalen) Beweisen zumeist stillschweigend genutzt werden:
 - Schließe von $P \wedge Q$ auf P .
 - Schließe von P und Q auf $P \wedge Q$.

Notwendigkeit formaler Beweise



Formale Beweise in Fitch

- Wir haben eine wohldefinierte Menge **formaler Beweisregeln**.
- Formale Beweise in Fitch können **mechanisch geprüft werden**.
- Für jeden Junktor gibt es
 - eine **Einführungsregel**, z. B. “von P schließe auf $P \vee Q$ ”,
 - eine **Beseitigungsregel**, z. B. “von $P \wedge Q$ schließe auf P ”.

Die einfachste Fitch-Regel: Reiteration

Reiteration (Reit):

$$\triangleright \left| \begin{array}{c} P \\ \vdots \\ P \end{array} \right.$$

Beweismethoden für Konjunktion und Disjunktion

Konjunktions-Beseitigung

Beweisregel (\wedge Elim)

Conjunction Elimination (\wedge Elim)

$$\frac{\begin{array}{c} P_1 \wedge \dots \wedge P_i \wedge \dots \wedge P_n \\ \vdots \\ P_i \end{array}}{\quad}$$

Konjunktions-Einführung

Beweisregel (\wedge Intro)

Conjunction Introduction (\wedge Intro)

$$\frac{\begin{array}{c} P_1 \\ \Downarrow \\ P_n \\ \vdots \\ \end{array}}{\triangleright P_1 \wedge \dots \wedge P_n}$$

Beispiel

$\text{Cube}(a) \wedge \text{Large}(a)$

$\text{Large}(a) \wedge \text{Cube}(a)$

Disjunktions-Einführung

Beweisregel (\vee Intro)

Disjunction Introduction (\vee Intro)

$$\triangleright \left| \begin{array}{c} P_i \\ \vdots \\ P_1 \vee \dots \vee P_i \vee \dots \vee P_n \end{array} \right.$$

Beispiel

$\text{Cube}(a) \wedge \text{Large}(a)$

$(\text{Cube}(a) \vee \text{Cube}(b)) \wedge (\text{Large}(a) \vee \text{Large}(b))$

Beweis durch Fallunterscheidung

Um auf S von $P_1 \vee \dots \vee P_n$ zu schließen, beweisen wir S , ausgehend von jedem einzelnen Disjunkt P_1, \dots, P_n .

Behauptung:

Angenommen, c ist entweder ein großer Würfel oder ein mittleres Tetraeder. Wir wollen zeigen: c ist nicht klein.

Beweis:

Fall 1: Falls c ein großer Würfel ist, ist c nicht klein.

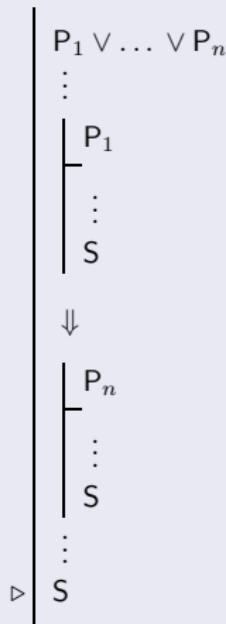
Fall 2: Falls c ein mittleres Tetraeder ist, ist c nicht klein.

Da dies nun in beiden Fällen gezeigt wurde, schließen wir, dass c nicht klein ist.

Disjunktions-Beseitigung

Beweisregel (\vee Elim)

Disjunction Elimination
(\vee Elim)



Beispiel

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

A

Eine inkorrekte Nutzung von Unterbeweisen

1. $(B \wedge A) \vee (A \wedge C)$
2. $B \wedge A$
3. B \wedge Elim: 2
4. A \wedge Elim: 2
5. $A \wedge C$
6. A \wedge Elim: 5
7. A \vee Elim: 1, 2–4, 5–6
8. $A \wedge B$ \wedge Intro: 7, 3

Die korrekte Nutzung von Unterbeweisen

- Um einen Beweisschritt in einem Unterbeweis zu überprüfen, kann man einen **vorangegangenen Schritt** im Hauptbeweis oder in jedem Unterbeweis, dessen Annahmen **noch gültig sind**, verwenden.
- Man darf **nie** einen Schritt aus einem Unterbeweis, der **bereits beendet** ist, verwenden.

Das Programm Fitch unterstützt das, indem es die Verwendung von einen Schritt aus einem Unterbeweis, der **bereits beendet** ist, verbietet.

Die korrekte Nutzung von Unterbeweisen: Beispiel

$$\vdash (A \wedge B) \vee (A \vee B)$$

$$B \vee A$$

Beweisregeln für Negation und Falsum

Beweis durch Widerspruch, indirekter Beweis

Um $\neg S$ zu zeigen, nehmen wir S an und zeigen einen Widerspruch \perp .
(\perp kann aus P und $\neg P$ abgeleitet werden.)

Wir nehmen $\text{Cube}(c) \vee \text{Dodec}(c)$ und $\text{Tet}(b)$ an.

Behauptung: $\neg(b = c)$.

Beweis: Angenommen, es sei $b = c$.

Fall 1: Wenn $\text{Cube}(c)$ gilt, dann gilt wegen $b = c$ auch $\text{Cube}(b)$, was einen Widerspruch zu $\text{Tet}(b)$ darstellt.

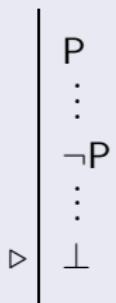
Fall 2: $\text{Dodec}(c)$ widerspricht analog $\text{Tet}(b)$.

In beiden Fällen erhalten wir einen Widerspruch, also kann unsere Annahme $b = c$ nicht richtig sein, folglich gilt $\neg(b = c)$.

\perp -Einführung

Beweisregel (\perp Intro)

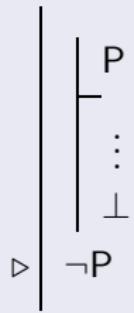
\perp Introduction
(\perp Intro)



Negations-Einführung

Beweisregel (\neg Intro)

Negation Introduction
(\neg Intro)



Beispiel

A

$\neg\neg A$

Negations-Beseitigung

Beweisregel (\neg Elim)

Negation Elimination (\neg Elim)

$$\begin{array}{c} \neg\neg P \\ \vdots \\ P \end{array}$$

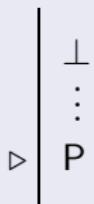
Beispiel



\perp -Beseitigung

Beweisregel (\perp Elim)

\perp Elimination
(\perp Elim)



Beispiel

$A \vee B$

$\neg B$

A

Zusammenfassung Veranstaltung 6

Zusammenfassung Veranstaltung 6 und Ausblick

Was haben wir heute gelernt?

- den Unterschied zwischen informalen und formalen Beweisen
- Fitch-Beweisregeln für Konjunktion, Disjunktion, Negation, Falsum

Nächstes Mal behandeln wir:

- Fitch-Beweisregeln für Konditional und Bikonditional
- Strategie und Taktik in Fitch
- Korrektheit und Vollständigkeit des Fitch-Kalküls

Lernziele Veranstaltung

7

Lernziele für heute (Veranstaltung 7)

- ich lerne Fitch-Beweisregeln für Konditional und Bikonditional kennen
- ich lerne Beweise für spezielle Argumente kennen
 - z.B. Argumente ohne Prämissen
- ich lerne Strategie und Taktik von Fitch-Beweisen
- ich lerne, was es heißt, dass das Fitch-Regelsystem korrekt ist
- ich lerne, was es heißt, dass das Fitch-Regelsystem vollständig ist, d.h. es genügend Beweisregeln gibt

Buch: 4.4, 5.4, 6.5, 6.6, 8.1-8.3, 17.2 (letzteres in Band 2)

Beweisregeln für (Bi-)Konditionale

Konditional-Beseitigung

Beweisregel (\rightarrow Elim)

Conditional Elimination (\rightarrow Elim)

$$\begin{array}{c} P \rightarrow Q \\ \vdots \\ P \\ \vdots \\ Q \end{array}$$

Konditional-Einführung

Beweisregel (\rightarrow Intro)

Conditional Introduction
(\rightarrow Intro)

$$\begin{array}{c} P \\ \vdash \quad : \\ Q \\ \hline \triangleright \quad P \rightarrow Q \end{array}$$

Beispiel

A → B

$$A \rightarrow (B \vee C)$$

Bikonditional-Beseitigung

Beweisregel (\leftrightarrow Elim)

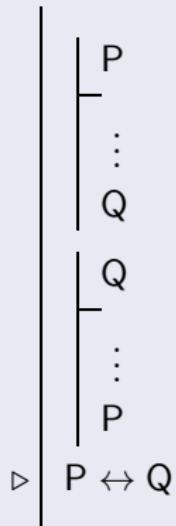
Biconditional Elimination (\leftrightarrow Elim)

\triangleright P \leftrightarrow Q (or Q \leftrightarrow P)
 :
 P
 :
 Q

Bikonditional-Einführung

Beweisregel (\leftrightarrow Intro)

Biconditional Introduction (\leftrightarrow Intro)



Beispiel

A ↔ B

$$B \wedge C$$

$$A \wedge C$$

Beweisstrategien für den Fitch-Kalkül

Ist dieses Argument gültig?

$\text{Home}(\text{max}) \vee \text{Home}(\text{claire})$

$\neg \text{Home}(\text{max})$

$\neg \text{Home}(\text{claire})$

$\text{Home}(\text{max}) \wedge \text{Happy}(\text{carl})$

Argumente mit sich widersprechenden Prämissen

Ein Beweis eines Widerspruchs \perp von Prämisen P_1, \dots, P_n (ohne zusätzliche Annahmen) bedeutet, dass die Prämismenmenge **inkonsistent** (und damit **unerfüllbar**) ist, die Prämisen sich also **widersprechen**. Ein Argument mit sich widersprechenden Prämisen ist **immer** gültig, aber (fast noch wichtiger) **niemals korrekt**.

Home(max) \vee Home(claire)

\neg Home(max)

\neg Home(claire)

Home(max) \wedge Happy(carl)

Argumente ohne Prämissen

Ein Beweis ohne jegliche Prämisse besagt, dass seine Konklusion eine **logische Wahrheit** ist.

Beispiel:

$$\begin{array}{c} | \\ \neg(P \wedge \neg P) \end{array}$$

Strategie und Taktik in Fitch

- ① Machen Sie **sich klar**, was die Sätze besagen.
- ② **Überlegen** Sie, ob die Konklusion aus den Prämissen folgt.
- ③ Wenn Sie meinen, dass die Konklusion nicht folgt oder sich nicht sicher sind, versuchen Sie, ein **Gegenbeispiel** zu finden.
- ④ Wenn Sie meinen, dass die Konklusion folgt, versuchen Sie, einen **informellen Beweis** zu führen.
- ⑤ Falls ein **formaler Beweis** gefordert ist, lassen Sie sich bei der Auffindung dessen **vom informellen Beweis** leiten.
- ⑥ Vergessen Sie nicht die Taktik des **Rückwärts-Arbeitens**, falls Sie formal oder informell nachweisen, dass ein Satz aus anderen folgt.
- ⑦ Wenn Sie rückwärts arbeiten, überprüfen Sie stets, ob die **Beweisziele Ihrer Zwischenschritte** aus den gegebenen Informationen folgen.

Strategie und Taktik in Fitch (Fortsetzung)

- Versuchen Sie stets, die Situation in Ihrem Beweis **mit den Beweisregeln zu vergleichen** (im Anhang des Buches gibt es die komplette Liste der verfügbaren Regeln, ebenso im internen Bereich auf der Webseite der Veranstaltung).
- Orientieren Sie sich in den Beweisschritten
 - am **Hauptjunktor einer der jeweiligen Prämissen** und wenden die entsprechende **Beseitigungsregel** an (vorwärts), oder
 - am **Hauptjunktor der jeweiligen Konklusion** und wenden die entsprechende **Einführungsregel** an (rückwärts).

Korrektheit des Fitch-Kalküls

Korrektheit und Vollständigkeit: Motivation

Wir haben zwei verschiedene Folgerungsbegriffe definiert:

- ① **logische (semantische) Folgerung**: die Konklusion ist wahr, wann immer die Prämissen wahr sind.
- ② **Beweisbarkeit**: die Konklusion lässt sich mit Hilfe des Fitchkalküls aus den Prämissen beweisen.

Wie verhalten sich diese beiden Begriffe zueinander?

Wiederholung: Aussagenlogische Sätze

Definition

Gegeben eine Menge \mathcal{A} von atomaren Sätzen, definieren wir die Menge der aussagenlogischen Sätze über \mathcal{A} induktiv wie folgt:

- ① jeder atomare Satz aus \mathcal{A} ist ein aussagenlogischer Satz;
 - ② \perp ist ein aussagenlogischer Satz;
 - ③ wenn P ein aussagenlogischer Satz ist, dann auch $\neg P$;
 - ④ wenn P_1 und P_2 aussagenlogische Sätze sind, dann auch $(P_1 \vee P_2)$, $(P_1 \wedge P_2)$, $(P_1 \rightarrow P_2)$ sowie $(P_1 \leftrightarrow P_2)$.

\perp , \neg , \vee , \wedge , \rightarrow , \leftrightarrow heißen dabei **Junktoren**.

Semantik aussagenlogischer Sätze

Wenn eine Belegung atomarer Sätze mit Wahrheitswerten gegeben ist, können wir diese mittels folgender [Wahrheitstafeln](#) auf komplexe Sätze erweitern:

P	Q	$P \rightarrow Q$	P	Q	$P \leftrightarrow Q$
T	T	T	T	T	T
T	F	F	T	F	F
F	T	T	F	T	F
F	F	T	F	F	T

\perp bedeutet immer F.

Beweisregeln

Konditional-Beseitigung
(\rightarrow Elim)

$$\frac{\begin{array}{c} P \rightarrow Q \\ \vdots \\ P \\ \vdots \\ \triangleright Q \end{array}}{\triangleright P \rightarrow Q}$$

Konditional-Einführung
(\rightarrow Intro)

$$\frac{\begin{array}{c} P \\ \vdots \\ Q \end{array}}{\triangleright P \rightarrow Q}$$

Disjunktions-Beseitigung
(\vee Elim)

$$\frac{\begin{array}{c} P_1 \vee \dots \vee P_n \\ \vdots \\ P_1 \\ \vdots \\ S \end{array}}{\Downarrow}$$

Negations-Einführung
(\neg Intro)

$$\frac{\begin{array}{c} P \\ \vdots \\ \perp \\ \triangleright P \end{array}}{\triangleright \neg P}$$

Bikonditional-Einführung
(\leftrightarrow Intro)

$$\frac{\begin{array}{c} P \\ \vdots \\ Q \\ \vdots \\ Q \\ \vdots \\ P \end{array}}{\triangleright P \leftrightarrow Q}$$

Konjunktions-Einführung
(\wedge Intro)

$$\frac{\begin{array}{c} P_1 \\ \Downarrow \\ P_n \\ \vdots \\ \triangleright P_1 \wedge \dots \wedge P_n \end{array}}{\triangleright P_i}$$

Konjunktions-Beseitigung
(\wedge Elim)

$$\frac{\begin{array}{c} P_1 \wedge \dots \wedge P_i \wedge \dots \wedge P_n \\ \vdots \\ P_i \end{array}}{\triangleright S}$$

Negations-Beseitigung
(\neg Elim)

$$\frac{\begin{array}{c} \neg P \\ \vdots \\ P \end{array}}{\triangleright \perp}$$

Bikonditional-Beseitigung
(\leftrightarrow Elim)

$$\frac{\begin{array}{c} P \leftrightarrow Q \text{ (oder } Q \leftrightarrow P) \\ \vdots \\ P \\ \vdots \\ Q \\ \vdots \\ \perp \end{array}}{\triangleright \perp}$$

Reiteration
(Reit)

$$\frac{\begin{array}{c} P \\ \vdots \\ P \end{array}}{\triangleright P_i \vee \dots \vee P_i \vee \dots \vee P_n}$$

\perp -Beseitigung
(\perp Elim)

$$\frac{\begin{array}{c} \perp \\ \vdots \\ P \end{array}}{\triangleright \perp}$$

\perp -Einführung
(\perp Intro)

$$\frac{\begin{array}{c} P \\ \vdots \\ \neg P \\ \vdots \\ \perp \end{array}}{\triangleright \perp}$$

Objekt- und Metatheorie

Objekttheorie = Beweise **innerhalb** eines formalen Beweissystems
(z. B. Fitch)

Metatheorie = Beweise **über** ein formales Beweissystem

Tautologische (semantische) Folgerung

Wir verallgemeinern den Begriff der tautologischen Folgerung auf beliebige (auch unendliche) Prämissenmengen:

Definition

Ein Satz S ist eine **tautologische Folgerung** aus einer logischen Theorie \mathcal{T} , geschrieben als

$$\mathcal{T} \models_{\mathcal{T}} S,$$

genau dann, wenn für alle Belegungen der atomaren Sätze mit Wahrheitswerten, die alle Sätze von \mathcal{T} wahr machen, auch S wahr ist.

Beispiel:

$$\{A \rightarrow B, B \rightarrow C\} \models_{\mathcal{T}} A \rightarrow C$$

Aussagenlogische Beweise (syntaktische Folgerung)

Definition

Ein Satz S ist $\mathcal{F}_{\mathcal{T}}$ -beweisbar aus einer logischen Theorie \mathcal{T} , in Zeichen

$$\mathcal{T} \vdash_T s,$$

wenn für S ein formaler Beweis (d.h. mit S als Zeile im Hauptbeweis) mit Prämissen aus \mathcal{T} existiert, der allein Reiteration und die Beseitigungs- und Einführungsregeln für $\vee, \wedge, \neg, \rightarrow, \leftrightarrow$ sowie \perp benutzt.

Auch hier kann \mathcal{T} beliebig, d.h. auch unendlich sein.

Beispiel:

$$\{A \rightarrow B, B \rightarrow C\} \vdash_T A \rightarrow C$$

Syntaktische versus semantische Folgerung

Syntaktische Folgerung \vdash_T	Semantische Folgerung \models_T																																																																						
<ul style="list-style-type: none"> ▶ 1. A \rightarrow B 2. B \rightarrow C 3. ∇A 4. B ✓ $\nabla \rightarrow$ Elim; 3,1 5. C ✓ $\nabla \rightarrow$ Elim; 4,2 6. A \rightarrow C ✓ $\nabla \rightarrow$ Intro; 3-5 	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>(1)</th><th>(2)</th><th>(3)</th><th></th><th>(1)</th><th>(2)</th><th>(3)</th></tr> <tr> <th>A</th><th>B</th><th>C</th><th></th><th>A \rightarrow B</th><th>B \rightarrow C</th><th>A \rightarrow C</th></tr> </thead> <tbody> <tr> <td>T</td><td>T</td><td>T</td><td></td><td>T</td><td>T</td><td>T</td></tr> <tr> <td>T</td><td>T</td><td>F</td><td></td><td>T</td><td>F</td><td>F</td></tr> <tr> <td>T</td><td>F</td><td>T</td><td></td><td>F</td><td>T</td><td>T</td></tr> <tr> <td>T</td><td>F</td><td>F</td><td></td><td>F</td><td>T</td><td>F</td></tr> <tr> <td>F</td><td>T</td><td>T</td><td></td><td>T</td><td>T</td><td>T</td></tr> <tr> <td>F</td><td>T</td><td>F</td><td></td><td>T</td><td>F</td><td>T</td></tr> <tr> <td>F</td><td>F</td><td>T</td><td></td><td>T</td><td>T</td><td>T</td></tr> <tr> <td>F</td><td>F</td><td>F</td><td></td><td>T</td><td>T</td><td>T</td></tr> </tbody> </table> <p style="text-align: center;">(1) (2) (3)</p>	(1)	(2)	(3)		(1)	(2)	(3)	A	B	C		A \rightarrow B	B \rightarrow C	A \rightarrow C	T	T	T		T	T	T	T	T	F		T	F	F	T	F	T		F	T	T	T	F	F		F	T	F	F	T	T		T	T	T	F	T	F		T	F	T	F	F	T		T	T	T	F	F	F		T	T	T
(1)	(2)	(3)		(1)	(2)	(3)																																																																	
A	B	C		A \rightarrow B	B \rightarrow C	A \rightarrow C																																																																	
T	T	T		T	T	T																																																																	
T	T	F		T	F	F																																																																	
T	F	T		F	T	T																																																																	
T	F	F		F	T	F																																																																	
F	T	T		T	T	T																																																																	
F	T	F		T	F	T																																																																	
F	F	T		T	T	T																																																																	
F	F	F		T	T	T																																																																	

Korrektheit

Theorem

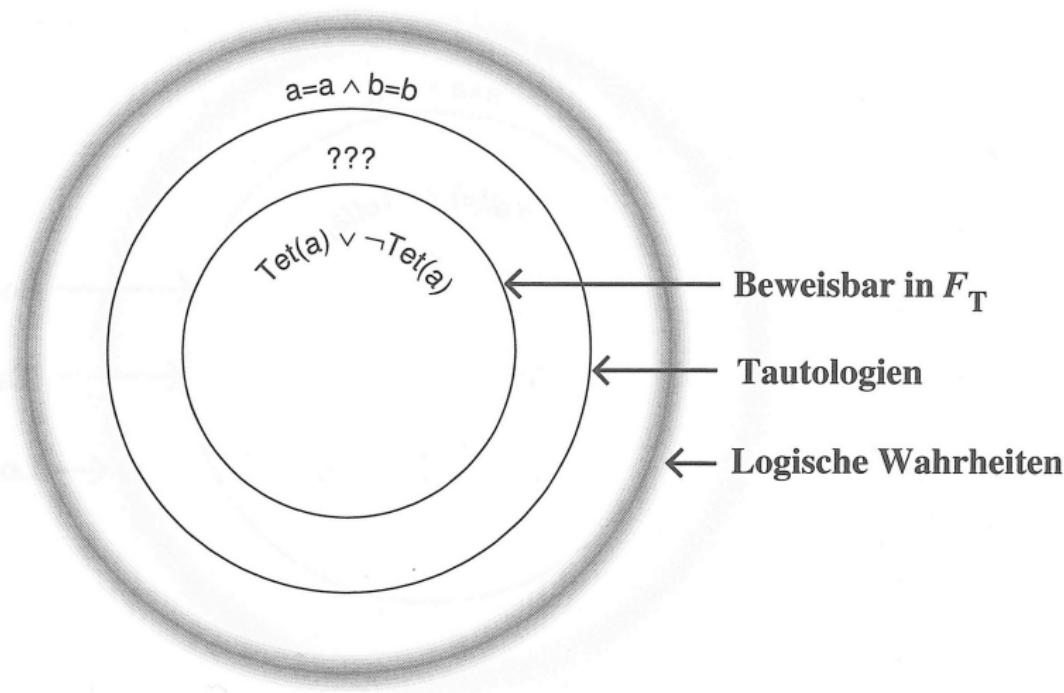
Das Beweissystem \mathcal{F}_T ist korrekt, d. h., wenn

$$\mathcal{T} \vdash_T S$$

gilt, dann auch

$$\mathcal{T} \vDash_T S.$$

Korrektheit



Korrektheit

Theorem

Das Beweissystem \mathcal{F}_T ist korrekt, d. h., wenn

$$\mathcal{T} \vdash_T S$$

gilt, dann auch

$$\mathcal{T} \vDash_T S.$$

Beweis.

Im Buch als Widerspruchsbeweis, unter Ausnutzung des ersten ungültigen Schrittes.

Hier durch starke Induktion über die Länge des Beweises (siehe folgende Folien).



Starke Induktion

Wir wollen zeigen, dass eine Aussage $A(n)$ gilt für alle $n \in \mathbb{N}$. Dazu reicht es, zu zeigen:

Falls

$A(k)$ für alle $k < m$ (*Induktionsvoraussetzung*),

dann

$A(m)$ (*Induktionsschluss*).

Wenn das gelingt, haben wir bewiesen: $A(n)$ für alle $n \in \mathbb{N}$.

Die Gültigkeit des Beweisprinzips der starken Induktion lässt sich mit Hilfe der normalen vollständigen Induktion beweisen.

Korrektheit

Das Korrektheits-Theorem folgt aus folgendem Lemma:

Lemma

Für $m \geq 1$ sei \mathcal{T}_m die Menge der Prämissen, die in Beweisschritt m in Kraft sind. Dann folgt der Satz in Zeile m logisch aus \mathcal{T}_m .

Beweis.

Induktionsschritt: Das Lemma gelte für die ersten m Zeilen eines Beweises. Wir betrachten den Satz in der $m + 1$ -ten Zeile. Falls sie eine Prämisse ist, ist sie sogar Element von \mathcal{T}_{m+1} . Falls nicht, wurde eine Regel angewandt. Wir machen eine Fallunterscheidung nach der angewandten Regel (siehe folgende Folien). □

Beweis-Fall: Disjunktions-Einführung

Disjunction Introduction (\vee Intro)

k	P_i ⋮
m+1	$P_1 \vee \dots \vee P_i \vee \dots \vee P_n$

Sei k die Zeile des Support-Steps P_i . Nach Induktionsvoraussetzung folgt P_i logisch aus $\mathcal{T}_k \subseteq \mathcal{T}_{m+1}$. Damit folgt auch $P_1 \vee \dots \vee P_n$ aus \mathcal{T}_{m+1} .

Beweis-Fall: Disjunktions-Beseitigung

Disjunction Elimination (\vee Elim)

d	$P_1 \vee \dots \vee P_n$ ⋮ P_1 ⋮ S
\downarrow	
s_1	P_n ⋮ S

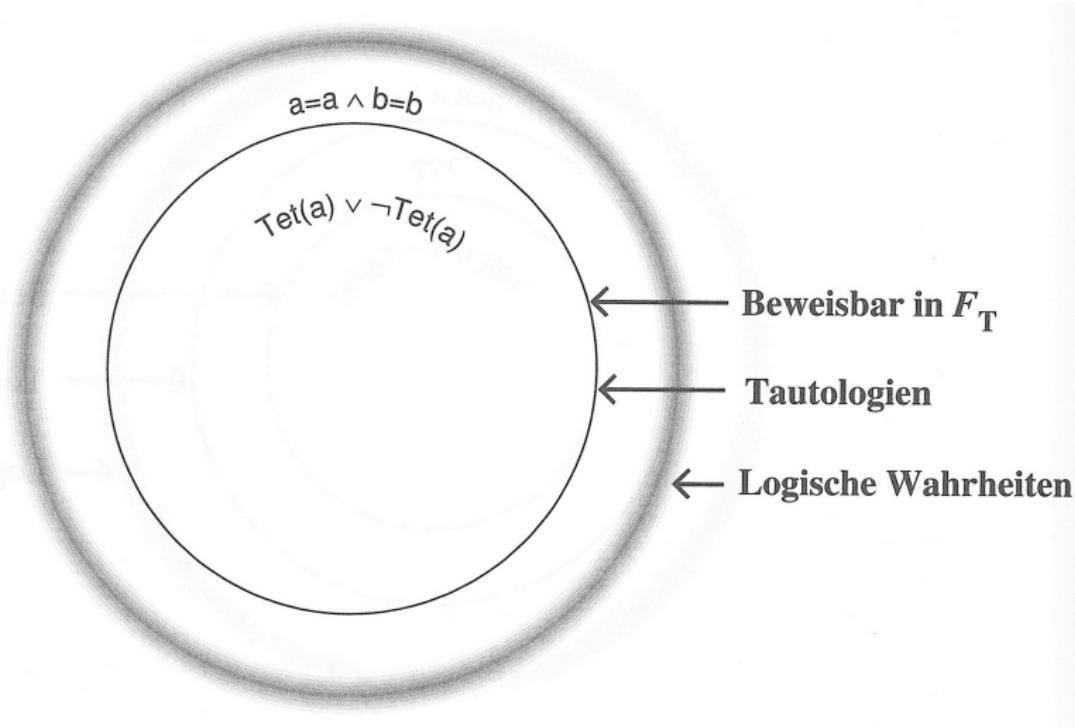
s_2	\vdots S
\downarrow	

$m+1$	\triangleright S
\downarrow	

Zur Vereinfachung nehmen wir an, dass $n = 2$. Seien $d, s_1, s_2, m + 1$ die Zeilennummern wie links angegeben. Nach Induktionsvoraussetzung ist $P_1 \vee P_2$ eine logische Folgerung aus $\mathcal{T}_d \subseteq \mathcal{T}_{m+1}$. Für $i = 1, 2$ ist zudem S eine logische Folgerung aus $\mathcal{T}_{s_i} \subseteq \mathcal{T}_{m+1} \cup \{P_i\}$. Eine Belegung, die \mathcal{T}_{m+1} wahr macht, muss daher $P_1 \vee P_2$ wahr machen, und somit auch eines der P_i , sagen wir P_{j_0} . Damit ist $\mathcal{T}_{s_{j_0}}$ wahr, und somit auch S .

Vollständigkeit des Fitch-Kalküls

Vollstndigkeit



Vollständigkeit der Aussagenlogik

Theorem 2 (Bernays, Post). Das Beweissystem \mathcal{F}_T ist vollständig, d. h., wenn

$$\mathcal{T} \vDash_T S$$

gilt, dann auch

$$\mathcal{T} \vdash_T S.$$

Tautologische Folgerung und Erfüllbarkeit

Satz 1. Der Satz S ist eine tautologische Folgerung aus der logischen Theorie \mathcal{T} genau dann, wenn die logische Theorie

$$\mathcal{T} \cup \{\neg S\}$$

nicht WT-erfüllbar ist.

Konsistenz

Eine logische Theorie \mathcal{T} heißt **formal inkonsistent**, wenn

$$\mathcal{T} \vdash_{\mathcal{T}} \perp$$

gilt, andernfalls heißt \mathcal{T} **formal konsistent**.

Beispiele:

$\{A \vee B, \neg A, \neg B\}$ ist formal inkonsistent.

$\{A \vee B, A, \neg B\}$ ist formal konsistent.

Vollständigkeit

Vollständigkeit-Satz (Bernays, Post). Das Beweissystem \mathcal{F}_T ist vollständig, d. h. wenn

$$\mathcal{T} \models_T S$$

gilt, dann auch

$$\mathcal{T} \vdash_T S.$$

Der Vollständigkeit-Satz folgt mit Satz 1 und Lemma 2 aus folgender Vollständigkeits-Variante:

Vollständigkeits-Variante. Jede formal konsistente logische Theorie ist WT-erfüllbar.

Lemma 2. $\mathcal{T} \cup \{\neg S\} \vdash_T \perp$ gilt genau dann, wenn $\mathcal{T} \vdash_T S$.

Beweis der Vollständigkeits-Variante

Eine logische Theorie \mathcal{T} heißt genau dann **formal vollständig**, wenn für jeden Satz S der Sprache entweder $\mathcal{T} \vdash_{\mathcal{T}} S$ oder $\mathcal{T} \vdash_{\mathcal{T}} \neg S$ gilt.

Satz 4. Jede formal vollständige und formal konsistente logische Theorie ist WT-erfüllbar.

Satz 6. Jede formal konsistente logische Theorie kann zu einer formal vollständigen und formal konsistenten logischen Theorie erweitert werden.

Beweis von Satz 4

Lemma 3. Es sei \mathcal{T} eine formal vollständige und formal konsistente logische Theorie. Dann gilt:

- ① $\mathcal{T} \not\vdash_{\mathcal{T}} \perp$
- ② $\mathcal{T} \vdash_{\mathcal{T}} (R \wedge S)$ genau dann, wenn $\mathcal{T} \vdash_{\mathcal{T}} R$ und $\mathcal{T} \vdash_{\mathcal{T}} S$.
- ③ $\mathcal{T} \vdash_{\mathcal{T}} (R \vee S)$ genau dann, wenn $\mathcal{T} \vdash_{\mathcal{T}} R$ oder $\mathcal{T} \vdash_{\mathcal{T}} S$.
- ④ $\mathcal{T} \vdash_{\mathcal{T}} (\neg S)$ genau dann, wenn $\mathcal{T} \not\vdash_{\mathcal{T}} S$.
- ⑤ $\mathcal{T} \vdash_{\mathcal{T}} (R \rightarrow S)$ genau dann, wenn $\mathcal{T} \not\vdash_{\mathcal{T}} R$ oder $\mathcal{T} \vdash_{\mathcal{T}} S$.
- ⑥ $\mathcal{T} \vdash_{\mathcal{T}} (R \leftrightarrow S)$ genau dann, wenn
 $(\mathcal{T} \vdash_{\mathcal{T}} R \text{ genau dann, wenn } \mathcal{T} \vdash_{\mathcal{T}} S).$

Satz 4. Jede formal vollständige und formal konsistente logische Theorie ist WT-erfüllbar.

Beweis von Satz 6

Lemma 5. Eine logische Theorie \mathcal{T} ist genau dann formal vollständig, wenn für jeden atomaren Satz A gilt, dass

$\mathcal{T} \vdash_T A$ oder $\mathcal{T} \vdash_T \neg A$.

Satz 6. Jede formal konsistente logische Theorie kann zu einer formal vollständigen und formal konsistenten logischen Theorie erweitert werden.

Beweis der Vollstndigkeit

- Angenommen, $\mathcal{T} \not\models_{\mathcal{T}} S$.
 - Nach Lemma 2 ist $\mathcal{T} \cup \{\neg S\}$ formal konsistent.
 - Diese logische Theorie kann nach Satz 4 und 6 zu einer formal konsistenten und formal vollständigen logischen Theorie erweitert werden, die WT-erfüllbar ist.
 - Sei h eine Bewertungsfunktion, die diese logische Theorie erfüllt.
 - Dann macht $h \mathcal{T}$ wahr, und S falsch. h ist also ein Gegenbeispiel zu $\mathcal{T} \models_{\mathcal{T}} S$.
 - Also $\mathcal{T} \not\models_{\mathcal{T}} S$.

Taut Con-Regel in Fitch

- Taut Con erlaubt, jede **tautologische Folgerung** in **einem Schritt** zu beweisen.
- Aufgrund des Vollständigkeitssatzes wissen wir, dass jede Anwendung von Taut Con durch **Regelanwendungen des aussagenlogischen Fitch-Kalküls ersetzt** werden kann.
- Da wir die aussagenlogischen Fitch-Regeln ausführlich geübt haben, darf in den Übungsaufgaben, in denen dies explizit angegeben ist, die Taut Con-Regel benutzt werden, sofern sie **einfache Schritte** betrifft, welche in einem informellen Beweis unerwähnt blieben würden.

Das sind insbesondere die bereits in der Veranstaltung vorgestellten **Äquivalenzen**.

Eigenschaften von Prädikaten in Bivalence World

Larger(a, b)	Cube(a)
Larger(b, c)	Tet(a)
Larger(a, c)	\perp
RightOf(b, c)	
LeftOf(c, b)	

Solche Argumente können in Fitch bewiesen werden durch Verwendung der besonderen Regel [Ana Con](#).

Argumente, die mit dieser Regel bewiesen wurden, sind im allgemeinen keine tautologischen Folgerungen, sondern nur Bivalence World-Folgerungen.

[Ana Con](#) ist sehr mächtig, sollte aber nur zum Beweis von Beziehungen zwischen atomaren Formeln (und ggf. \perp) verwendet werden.

Kompaktheits-Theorem

Komplettetheorem. Es sei \mathcal{T} eine logische Theorie. Wenn jede endliche Teilmenge von \mathcal{T} WT-erfüllbar ist, so ist auch \mathcal{T} WT-erfüllbar.

Zusammenfassung Veranstaltung 7

Zusammenfassung Veranstaltung 7 und Ausblick

Was haben wir heute gelernt?

- Beweisregeln für das (Bi-)Konditional
 - Beweise für spezielle Argumente
 - Argumente mit sich widersprechenden Prämissen
 - diese sind immer gültig und beweisbar
 - Argumente ohne Prämissen
 - Strategie und Taktik von Fitch-Beweisen
 - vorwärts beweisen: eliminiere den Hauptjunktor einer Prämisse oder schon bewiesenen Zeile
 - rückwärts beweisen: führe den Hauptjunktor einer Prämisse oder schon bewiesenen Zeile ein
 - Korrektheit und Vollständigkeit des Fitch-Kalküls

Nächstes Mal behandeln wir:

- Effiziente Algorithmen für logische Probleme:
 - SAT-Solving (für beliebige Sätze in KNF)
 - Erfüllbarkeitsalgorithmus für Hornformeln

Lernziele Veranstaltung 8

Lernziele für heute (Veranstaltung 8)

Ich lerne Algorithmen kennen, um tautologische Erfüllbarkeit zu prüfen:

- SAT-Solving, eine effiziente und in der Praxis häufig angewandte Methode, um tautologische Erfüllbarkeit zu prüfen
- den Erfüllbarkeits-Algorithmus für Hornformeln.

Buch: 17.3 (Band 2)

Uwe Schöning, Jacobo Torán: Das Erfüllbarkeitsproblem SAT — Algorithmen und Analysen. Lehmanns Verlag 2012.

SAT-Solving

Wiederholung: Konjunktive Normalform (KNF)

Für jeden aussagenlogischen Satz gibt es einen äquivalenten Satz in konjunktiver Normalform (KNF), d. h. einen Satz der Form

$$(\varphi_{1,1} \vee \cdots \vee \varphi_{1,m_1}) \wedge \cdots \wedge (\varphi_{n,1} \vee \cdots \vee \varphi_{n,m_n}) \quad (n \geq 1, m_i \geq 1),$$

wobei $\varphi_{i,j}$ Literale sind, d. h. atomare Sätze oder negierte atomare Sätze.

Man beachte, dass n auch 1 sein kann, z. B. ist $A \vee B$ in KNF.

Man beachte, dass m_i auch 1 sein kann, z. B. sind sowohl A als auch $A \wedge B$ in KNF.

Die Konjunktionsglieder ($\varphi_{i,1} \vee \dots \vee \varphi_{i,m_i}$) heißen auch **Klauseln**.

Erfüllbarkeit und logische Folgerung

Logische Folgerungen können auf (Un-)Erfüllbarkeit zurückgeführt werden:

Theorem

Die *tautologische Folgerung* $\mathcal{T} \models_{\mathcal{T}} S$ gilt

genau dann, wenn

$\mathcal{T} \cup \{\neg S\}$ *WT-unerfüllbar* ist.

Weitere Anwendungen tautologischer Erfüllbarkeit (SAT)

- Schaltkreisentwurf
- Stundenplanung
- Packungsprobleme
- Tourenplanung
- ...

SAT-Solving prüft tautologische Erfüllbarkeit.

SAT-Solving mit DPLL für KNF

Davis-Putnam-Logemann-Loveland-Algorithmus (DPLL)

- Backtracking-Algorithmus:

- 1 wähle ein Literal,
 - 2 weise ihm einen Wahrheitswert zu,
 - 3 vereinfache den Satz: für jede Klausel
 - lösche alle falschen Literale. Wenn keines mehr übrig bleibt, ersetze das letzte durch \perp und gebe „unerfüllbar“ zurück
 - lösche die Klausel, falls ein Literal wahr ist. Wenn keine Klausel mehr übrig bleibt, ersetze die letzte durch \top und gebe „erfüllbar“ zurück
 - 4 prüfe rekursiv, ob der vereinfachte Satz erfüllbar ist,
 - wenn das der Fall ist, ist der ursprüngliche Satz erfüllbar;
 - andernfalls mache Backtracking zu Schritt 2 und wähle dort den anderen Wahrheitswert.

DPLL: Beispiel 1

Ist $(\neg A \vee B) \wedge (\neg B \vee \neg A) \wedge (A \vee C)$ erfüllbar?

DPLL: Beispiel 2

Ist $(\neg A \vee B) \wedge (\neg B \vee \neg A) \wedge (A \vee C) \wedge (\neg C \vee A)$ erfüllbar?

Optimierungen in DPLL

- Wenn eine Klausel eine **Einheitsklausel** ist, d. h. wenn sie nur ein einziges, noch mit keinem Wahrheitswert belegten Literal enthält, dann kann diese Klausel nur mit einem Wahrheitswert so belegt werden, dass sie wahr wird. Die vorrangige Wahl von Einheitsklauseln kann also den Suchraum reduzieren.
- **Elimination reiner Literale:** Wenn ein atomarer Satz nur mit einer Polarität (d. h. nur unnegiert oder nur negiert) in dem Satz vorkommt, wird sie **rein** genannt. Die Wahrheitswert-Belegung ist dann klar:
 - wahr, wenn der atomare Satz nur unnegiert vorkommt,
 - falsch, wenn der atomare Satz nur negiert vorkommt.

DPLL: Beispiel 2

Ist $(\neg A \vee B) \wedge (\neg B \vee \neg A) \wedge (A \vee C) \wedge (\neg C \vee A)$ erfüllbar?

DIMACS-Format

c SAT-Testbeispiel	Kommentar
p cnf 3 2	Problem in KNF, 3 atomare Sätze, 2 Sätze
1 -3 0	$A_1 \vee \neg A_3$ – ist Negation
2 3 -1 0	$A_2 \vee A_3 \vee \neg A_1$ 0 beendet die Klausel

Ausgabe von Minisat

SAT 1 2 -3 A_1, A_2 wahr, A_3 falsch

Minisat download: <http://minisat.se>

Minisat online: <https://tinyurl.com/minisat-online>

Spezifikationssprache DOL

- standardisiert durch Object Management Group (OMG)
- gut lesbare Syntax für Aussagenlogik

```
logic Propositional
spec Props =
    props A,B,C
        . A
        . not (A /\ B)
        . C => B
        . not C %implied
    end
```

Heterogeneous Tool Set

- Kann DOL-Dokumente einlesen und prüfen
- Kann Sätze in KNF umwandeln
- Kann mit %implied markierte Sätze beweisen
(mittels SAT-Solvern)
 - „Prove“-Menü eines Knoten verwenden
- Kann Belegungen für Sätze und Mengen von Sätzen finden
(mittels SAT-Solvern)
 - „Check consistency“-Menü eines Knoten verwenden
- Hets ist verfügbar unter <http://hets.eu>
- Online-Version unter <http://rest.hets.eu>

Hornformeln

Hornformeln

Definition (Hornformel)

Ein Satz in KNF heißt **Hornformel**, wenn in dem Satz jede Disjunktion von Literalen **höchstens ein positives Literal**, d. h. einen nichtnegierten atomaren Satz, enthält.

Beispiele für Hornformeln

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

$$A \wedge B \wedge \neg C$$

$$A \vee \neg B \vee \neg C$$

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

Beispiele für Nicht-Hornformeln

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

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

$$A \vee (B \vee \neg D)$$

Alternative Darstellung der Disjunktionen in Hornformeln

$$\neg A_1 \vee \cdots \vee \neg A_n \vee B \quad \Leftrightarrow \quad (A_1 \wedge \cdots \wedge A_n) \rightarrow B$$

$$\neg A_1 \vee \cdots \vee \neg A_n \quad \Leftrightarrow \quad (A_1 \wedge \cdots \wedge A_n) \rightarrow \perp$$

$$B \qquad \Leftrightarrow \qquad \top \rightarrow B$$

Jede Hornformel ist äquivalent zu einer Konjunktion von Implikationen der obigen drei Formen.

\top (verum) ist eine Abkürzung für $\neg\perp$ und damit ein Satz, der immer wahr ist.

Erfüllbarkeitsalgorithmus für Hornformeln

Es sei S eine Hornformel in konditionaler Form, die aus atomaren Sätzen sowie aus \top und \perp besteht.

- ① Für jede Implikation der Form $\top \rightarrow B$ nimm B in die Menge \mathcal{T} auf.
- ② Falls für eine Implikation der Form $(A_1 \wedge \dots \wedge A_n) \rightarrow B$ die atomaren Sätze A_1, \dots, A_n bereits in \mathcal{T} enthalten ist, so füge auch B zu \mathcal{T} hinzu.
- ③ Wiederhole Schritt 2 so lange als möglich.
- ④ Falls $\perp \in \mathcal{T}$, ist S nicht WT-erfüllbar.

Andernfalls belege alle atomaren Sätze in \mathcal{T} mit wahr, und die anderen mit falsch. Die so erhaltene Belegung macht S wahr.

Achtung: die DPLL-Optimierungen auf Folie 246 sind hier nicht vorgesehen!

Erfüllbarkeitsalgorithmus für Hornformeln — Beispiel

Da sich die Menge \mathcal{T} im Verlauf ändert, verwenden wir die Bezeichnungen $\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3, \dots$

$$(A \wedge B \wedge G) \rightarrow E$$

$$(B \wedge C \wedge E) \rightarrow \perp$$

$$(A \wedge B \wedge C \wedge D) \rightarrow \perp$$

$$\top \rightarrow A$$

$$(A \wedge B) \rightarrow D$$

$$(A \wedge E) \rightarrow C$$

$$A \rightarrow C$$

$$(A \wedge C) \rightarrow B$$

Erfüllbarkeitsalgorithmus für Hornformeln — Beispiel

Da sich die Menge \mathcal{T} im Verlauf ändert, verwenden wir die Bezeichnungen $\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3, \dots$

$$(A \wedge B \wedge G) \rightarrow E \quad \text{Schritt 1}$$

$$(B \wedge C \wedge E) \rightarrow \perp \quad \mathcal{T}_1 = \{A\}$$

$$(A \wedge B \wedge C \wedge D) \rightarrow \perp$$

$$\top \rightarrow A$$

$$(A \wedge B) \rightarrow D$$

$$(A \wedge E) \rightarrow C$$

$$A \rightarrow C$$

$$(A \wedge C) \rightarrow B$$

Erfüllbarkeitsalgorithmus für Hornformeln — Beispiel

Da sich die Menge \mathcal{T} im Verlauf ändert, verwenden wir die Bezeichnungen $\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3, \dots$

- | | |
|--|-------------------------|
| $(\boxed{A} \wedge B \wedge G) \rightarrow E$ | Schritt 1 |
| $(B \wedge C \wedge E) \rightarrow \perp$ | $\mathcal{T}_1 = \{A\}$ |
| $(\boxed{A} \wedge B \wedge C \wedge D) \rightarrow \perp$ | |
| $\top \rightarrow \boxed{A}$ | |
| $(\boxed{A} \wedge B) \rightarrow D$ | |
| $(\boxed{A} \wedge E) \rightarrow C$ | |
| $\boxed{A} \rightarrow C$ | |
| $(\boxed{A} \wedge C) \rightarrow B$ | |

Erfüllbarkeitsalgorithmus für Hornformeln — Beispiel

Da sich die Menge \mathcal{T} im Verlauf ändert, verwenden wir die Bezeichnungen $\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3, \dots$

$(A \wedge B \wedge G) \rightarrow E$	Schritt 1
$(B \wedge C \wedge E) \rightarrow \perp$	$\mathcal{T}_1 = \{A\}$
$(A \wedge B \wedge C \wedge D) \rightarrow \perp$	Schritte 2+3
$T \rightarrow A$	$\mathcal{T}_2 = \{A, C\}$
$(A \wedge B) \rightarrow D$	
$(A \wedge E) \rightarrow C$	
$A \rightarrow C$	
$(A \wedge C) \rightarrow B$	

Erfüllbarkeitsalgorithmus für Hornformeln — Beispiel

Da sich die Menge \mathcal{T} im Verlauf ändert, verwenden wir die Bezeichnungen $\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3, \dots$

$(A \wedge B \wedge G) \rightarrow E$	Schritt 1
$(B \wedge C \wedge E) \rightarrow \perp$	$\mathcal{T}_1 = \{A\}$
$(A \wedge B \wedge C \wedge D) \rightarrow \perp$	Schritte 2+3
$T \rightarrow A$	$\mathcal{T}_2 = \{A, C\}$
$(A \wedge B) \rightarrow D$	
$(A \wedge E) \rightarrow C$	
$A \rightarrow C$	
$(A \wedge C) \rightarrow B$	

Erfüllbarkeitsalgorithmus für Hornformeln — Beispiel

Da sich die Menge \mathcal{T} im Verlauf ändert, verwenden wir die Bezeichnungen $\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3, \dots$

$(A \wedge B \wedge G) \rightarrow E$	Schritt 1
$(B \wedge C \wedge E) \rightarrow \perp$	$\mathcal{T}_1 = \{A\}$
$(A \wedge B \wedge C \wedge D) \rightarrow \perp$	Schritte 2+3
$T \rightarrow A$	$\mathcal{T}_2 = \{A, C\}$
$(A \wedge B) \rightarrow D$	$\mathcal{T}_3 = \{A, C, B\}$
$(A \wedge E) \rightarrow C$	
$A \rightarrow C$	
$(A \wedge C) \rightarrow B$	

Erfüllbarkeitsalgorithmus für Hornformeln — Beispiel

Da sich die Menge \mathcal{T} im Verlauf ändert, verwenden wir die Bezeichnungen $\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3, \dots$

$$(A \wedge B \wedge G) \rightarrow E$$

Schritt 1

$$(B \wedge C \wedge E) \rightarrow \perp$$

$$\mathcal{T}_1 = \{A\}$$

$$(A \wedge B \wedge C \wedge D) \rightarrow \perp$$

Schritte 2+3

$$T \rightarrow A$$

$$\mathcal{T}_2 = \{A, C\}$$

$$(A \wedge B) \rightarrow D$$

$$\mathcal{T}_3 = \{A, C, B\}$$

$$(A \wedge E) \rightarrow C$$

$$A \rightarrow C$$

$$(A \wedge C) \rightarrow B$$

Erfüllbarkeitsalgorithmus für Hornformeln — Beispiel

Da sich die Menge \mathcal{T} im Verlauf ändert, verwenden wir die Bezeichnungen $\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3, \dots$

$$(A \wedge B \wedge G) \rightarrow E$$

Schritt 1

$$(B \wedge C \wedge E) \rightarrow \perp$$

$$\mathcal{T}_1 = \{A\}$$

$$(A \wedge B \wedge C \wedge D) \rightarrow \perp$$

Schritte 2+3

$$\top \rightarrow A$$

$$\mathcal{T}_2 = \{A, C\}$$

$$(A \wedge B) \rightarrow D$$

$$\mathcal{T}_3 = \{A, C, B\}$$

$$(A \wedge E) \rightarrow C$$

$$\mathcal{T}_4 = \{A, C, B, D\}$$

$$A \rightarrow C$$

$$(A \wedge C) \rightarrow B$$

Erfüllbarkeitsalgorithmus für Hornformeln — Beispiel

Da sich die Menge \mathcal{T} im Verlauf ändert, verwenden wir die Bezeichnungen $\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3, \dots$

$$(A \wedge B \wedge G) \rightarrow E$$

Schritt 1

$$(B \wedge C \wedge E) \rightarrow \perp$$

$$\mathcal{T}_1 = \{A\}$$

$$(A \wedge B \wedge C \wedge D) \rightarrow \perp$$

Schritte 2+3

$$T \rightarrow A$$

$$\mathcal{T}_2 = \{A, C\}$$

$$(A \wedge B) \rightarrow D$$

$$\mathcal{T}_3 = \{A, C, B\}$$

$$(A \wedge E) \rightarrow C$$

$$\mathcal{T}_4 = \{A, C, B, D\}$$

$$A \rightarrow C$$

$$(A \wedge C) \rightarrow B$$

Erfüllbarkeitsalgorithmus für Hornformeln — Beispiel

Da sich die Menge \mathcal{T} im Verlauf ändert, verwenden wir die Bezeichnungen $\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3, \dots$

$(A \wedge B \wedge G) \rightarrow E$	Schritt 1
$(B \wedge C \wedge E) \rightarrow \perp$	$\mathcal{T}_1 = \{A\}$
$(A \wedge B \wedge C \wedge D) \rightarrow \perp$	Schritte 2+3
$\top \rightarrow A$	$\mathcal{T}_2 = \{A, C\}$
$(A \wedge B) \rightarrow D$	$\mathcal{T}_3 = \{A, C, B\}$
$(A \wedge E) \rightarrow C$	$\mathcal{T}_4 = \{A, C, B, D\}$
$A \rightarrow C$	$\mathcal{T}_5 = \{A, C, B, D, \perp\}$
$(A \wedge C) \rightarrow B$	

Erfüllbarkeitsalgorithmus für Hornformeln — Beispiel

Da sich die Menge \mathcal{T} im Verlauf ändert, verwenden wir die Bezeichnungen $\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3, \dots$

$(A \wedge B \wedge G) \rightarrow E$	Schritt 1
$(B \wedge C \wedge E) \rightarrow \perp$	$\mathcal{T}_1 = \{A\}$
$(A \wedge B \wedge C \wedge D) \rightarrow \perp$	Schritte 2+3
$T \rightarrow A$	$\mathcal{T}_2 = \{A, C\}$
$(A \wedge B) \rightarrow D$	$\mathcal{T}_3 = \{A, C, B\}$
$(A \wedge E) \rightarrow C$	$\mathcal{T}_4 = \{A, C, B, D\}$
$A \rightarrow C$	$\mathcal{T}_5 = \{A, C, B, D, \perp\}$
$(A \wedge C) \rightarrow B$	

Schritt 4: Unerfüllbar!

Erfüllbarkeitsalgorithmus für Hornformeln — Beispiel

$$(A \wedge B \wedge G) \rightarrow E$$

$$(B \wedge C \wedge E) \rightarrow \perp$$

$$(A \wedge B \wedge C \wedge D) \rightarrow \perp$$

$$\top \rightarrow A$$

$$(A \wedge B \wedge E) \rightarrow D$$

$$(A \wedge E) \rightarrow C$$

$$A \rightarrow C$$

$$(A \wedge C) \rightarrow B$$

Erfüllbarkeitsalgorithmus für Hornformeln — Beispiel

$$(A \wedge B \wedge G) \rightarrow E$$

$$(B \wedge C \wedge E) \rightarrow \perp$$

$$(A \wedge B \wedge C \wedge D) \rightarrow \perp$$

$$\top \rightarrow A$$

$$(A \wedge B \wedge E) \rightarrow D$$

$$(A \wedge E) \rightarrow C$$

$$A \rightarrow C$$

$$(A \wedge C) \rightarrow B$$

Schritt 1

$$\mathcal{T}_1 = \{A\}$$

Erfüllbarkeitsalgorithmus für Hornformeln — Beispiel

$$(A \wedge B \wedge G) \rightarrow E$$

$$(B \wedge C \wedge E) \rightarrow \perp$$

$$(A \wedge B \wedge C \wedge D) \rightarrow \perp$$

$$\top \rightarrow A$$

$$(A \wedge B \wedge E) \rightarrow D$$

$$(A \wedge E) \rightarrow C$$

$$A \rightarrow C$$

$$(A \wedge C) \rightarrow B$$

Schritt 1

$$\mathcal{T}_1 = \{A\}$$

Erfüllbarkeitsalgorithmus für Hornformeln — Beispiel

$$(A \wedge B \wedge G) \rightarrow E$$

$$(B \wedge C \wedge E) \rightarrow \perp$$

$$(A \wedge B \wedge C \wedge D) \rightarrow \perp$$

$$\top \rightarrow A$$

$$(A \wedge B \wedge E) \rightarrow D$$

$$(A \wedge E) \rightarrow C$$

$$A \rightarrow C$$

$$(A \wedge C) \rightarrow B$$

Schritt 1

$$\mathcal{T}_1 = \{A\}$$

Schritte 2+3

$$\mathcal{T}_2 = \{A, C\}$$

Erfüllbarkeitsalgorithmus für Hornformeln — Beispiel

$$(A \wedge B \wedge G) \rightarrow E$$

$$(B \wedge C \wedge E) \rightarrow \perp$$

$$(A \wedge B \wedge C \wedge D) \rightarrow \perp$$

$$\top \rightarrow A$$

$$(A \wedge B \wedge E) \rightarrow D$$

$$(A \wedge E) \rightarrow C$$

$$A \rightarrow C$$

$$(A \wedge C) \rightarrow B$$

Schritt 1

$$\mathcal{T}_1 = \{A\}$$

Schritte 2+3

$$\mathcal{T}_2 = \{A, C\}$$

Erfüllbarkeitsalgorithmus für Hornformeln — Beispiel

$$(A \wedge B \wedge G) \rightarrow E$$

$$(B \wedge C \wedge E) \rightarrow \perp$$

$$(A \wedge B \wedge C \wedge D) \rightarrow \perp$$

$$\top \rightarrow A$$

$$(A \wedge B \wedge E) \rightarrow D$$

$$(A \wedge E) \rightarrow C$$

$$A \rightarrow C$$

$$(A \wedge C) \rightarrow B$$

Schritt 1

$$\mathcal{T}_1 = \{A\}$$

Schritte 2+3

$$\mathcal{T}_2 = \{A, C\}$$

$$\mathcal{T}_3 = \{A, C, B\}$$

Erfüllbarkeitsalgorithmus für Hornformeln — Beispiel

$$(A \wedge B \wedge G) \rightarrow E$$

$$(B \wedge C \wedge E) \rightarrow \perp$$

$$(A \wedge B \wedge C \wedge D) \rightarrow \perp$$

$$\top \rightarrow A$$

$$(A \wedge B \wedge E) \rightarrow D$$

$$(A \wedge E) \rightarrow C$$

$$A \rightarrow C$$

$$(A \wedge C) \rightarrow B$$

Schritt 1

$$\mathcal{T}_1 = \{A\}$$

Schritte 2+3

$$\mathcal{T}_2 = \{A, C\}$$

$$\mathcal{T}_3 = \{A, C, B\}$$

Erfüllbarkeitsalgorithmus für Hornformeln — Beispiel

$$\begin{aligned}
 & (\boxed{A} \wedge \boxed{B} \wedge G) \rightarrow E \\
 & (\boxed{B} \wedge \boxed{C} \wedge E) \rightarrow \perp \\
 & (\boxed{A} \wedge \boxed{B} \wedge \boxed{C} \wedge D) \rightarrow \perp \\
 & T \rightarrow \boxed{A} \\
 & (\boxed{A} \wedge \boxed{B} \wedge E) \rightarrow D \\
 & (\boxed{A} \wedge E) \rightarrow C \\
 & \boxed{A} \rightarrow \boxed{C} \\
 & (\boxed{A} \wedge \boxed{C}) \rightarrow B
 \end{aligned}$$

Schritt 1

$$\mathcal{T}_1 = \{A\}$$

Schritte 2+3

$$\mathcal{T}_2 = \{A, C\}$$

$$\mathcal{T}_3 = \{A, C, B\}$$

Schritt 4: Erfüllbar!

Erfüllende Belegung: $h(P) = T$ gdw $P \in \mathcal{T}_3$

Erfüllbarkeitsalgorithmus für Hornformeln — Beispiel

$$\begin{aligned} & (\boxed{A} \wedge \boxed{B} \wedge G) \rightarrow E \\ & (\boxed{B} \wedge \boxed{C} \wedge E) \rightarrow \perp \\ & (\boxed{A} \wedge \boxed{B} \wedge \boxed{C} \wedge D) \rightarrow \perp \\ & \top \rightarrow \boxed{A} \\ & (\boxed{A} \wedge \boxed{B} \wedge E) \rightarrow D \\ & (\boxed{A} \wedge E) \rightarrow C \\ & \boxed{A} \rightarrow \boxed{C} \\ & (\boxed{A} \wedge \boxed{C}) \rightarrow B \end{aligned}$$

Schritt 1

$$\mathcal{T}_1 = \{A\}$$

Schritte 2+3

$$\mathcal{T}_2 = \{A, C\}$$

$$\mathcal{T}_3 = \{A, C, B\}$$

Schritt 4: Erfüllbar!

Erfüllende Belegung: $h(P) = T$ gdw $P \in \mathcal{T}_3$

$$h(A) = h(B) = h(C) = T,$$

$$h(D) = h(E) = h(G) = F.$$

Erfüllbarkeitsalgorithmus für Hornformeln — Beispiel

$$\begin{aligned}(A \wedge B \wedge G) &\rightarrow E \\(B \wedge C \wedge E) &\rightarrow \perp \\(A \wedge B \wedge C \wedge D) &\rightarrow \perp \\(A \wedge B) &\rightarrow D \\(A \wedge E) &\rightarrow C \\A &\rightarrow C \\(A \wedge C) &\rightarrow B\end{aligned}$$

Erfüllbarkeitsalgorithmus für Hornformeln — Beispiel

$$(A \wedge B \wedge G) \rightarrow E$$

$$(B \wedge C \wedge E) \rightarrow \perp$$

$$(A \wedge B \wedge C \wedge D) \rightarrow \perp$$

$$(A \wedge B) \rightarrow D$$

$$(A \wedge E) \rightarrow C$$

$$A \rightarrow C$$

$$(A \wedge C) \rightarrow B$$

Schritt 1

$$\mathcal{T}_1 = \emptyset$$

Erfüllbarkeitsalgorithmus für Hornformeln — Beispiel

$$(A \wedge B \wedge G) \rightarrow E$$

$$(B \wedge C \wedge E) \rightarrow \perp$$

$$(A \wedge B \wedge C \wedge D) \rightarrow \perp$$

$$(A \wedge B) \rightarrow D$$

$$(A \wedge E) \rightarrow C$$

$$A \rightarrow C$$

$$(A \wedge C) \rightarrow B$$

Schritt 1

$$\mathcal{T}_1 = \emptyset$$

Schritte 2+3

$$\mathcal{T}_2 = \emptyset$$

Erfüllbarkeitsalgorithmus für Hornformeln — Beispiel

$$(A \wedge B \wedge G) \rightarrow E$$

$$(B \wedge C \wedge E) \rightarrow \perp$$

$$(A \wedge B \wedge C \wedge D) \rightarrow \perp$$

$$(A \wedge B) \rightarrow D$$

$$(A \wedge E) \rightarrow C$$

$$A \rightarrow C$$

$$(A \wedge C) \rightarrow B$$

Schritt 1

$$\mathcal{T}_1 = \emptyset$$

Schritte 2+3

$$\mathcal{T}_2 = \emptyset$$

Schritt 4: Erfüllbar!

Erfüllende Belegung: $h(P) = T$ gdw $P \in \mathcal{T}_2$

Erfüllbarkeitsalgorithmus für Hornformeln — Beispiel

$$(A \wedge B \wedge G) \rightarrow E$$

$$(B \wedge C \wedge E) \rightarrow \perp$$

$$(A \wedge B \wedge C \wedge D) \rightarrow \perp$$

$$(A \wedge B) \rightarrow D$$

$$(A \wedge E) \rightarrow C$$

$$A \rightarrow C$$

$$(A \wedge C) \rightarrow B$$

Schritt 1

$$\mathcal{T}_1 = \emptyset$$

Schritte 2+3

$$\mathcal{T}_2 = \emptyset$$

Schritt 4: Erfüllbar!

Erfüllende Belegung: $h(P) = T$ gdw $P \in \mathcal{T}_2$

$$h(A) = h(B) = h(C) = h(D) = h(E) = h(G) = F.$$

Triviale Erfüllbarkeit von Hornformeln — Beispiel

$$(A \wedge B \wedge G) \rightarrow E$$

$$\top \rightarrow A$$

$$(A \wedge B) \rightarrow D$$

$$(A \wedge E) \rightarrow C$$

$$A \rightarrow C$$

$$(A \wedge C) \rightarrow B$$

Triviale Erfüllbarkeit von Hornformeln — Beispiel

$$(A \wedge B \wedge G) \rightarrow E$$

$$\top \rightarrow A$$

$$(A \wedge B) \rightarrow D$$

$$(A \wedge E) \rightarrow C$$

$$A \rightarrow C$$

$$(A \wedge C) \rightarrow B$$

Erfüllbar!

Erfüllende Belegung laut Algorithmus:

$$h(A) = h(B) = h(C) = h(D) = T, \quad h(E) = h(G) = F.$$

Triviale Erfüllbarkeit von Hornformeln — Beispiel

$$\begin{aligned}
 & (A \wedge B \wedge G) \rightarrow E \\
 & \top \rightarrow A \\
 & (A \wedge B) \rightarrow D \\
 & (A \wedge E) \rightarrow C \\
 & A \rightarrow C \\
 & (A \wedge C) \rightarrow B
 \end{aligned}$$

Erfüllbar!

Erfüllende Belegung laut Algorithmus:

$$h(A) = h(B) = h(C) = h(D) = T, \quad h(E) = h(G) = F.$$

Triviale erfüllende Belegung:

$$h(A) = h(B) = h(C) = h(D) = h(E) = h(G) = T.$$

Erfüllbarkeitsalgorithmus für Hornformeln

Es sei S eine Hornformel in konditionaler Form, die aus atomaren Sätzen sowie aus \top und \perp besteht.

- ① Für jede Implikation der Form $\top \rightarrow B$ nimm B in die Menge \mathcal{T} auf.
- ② Falls für eine Implikation der Form $(A_1 \wedge \dots \wedge A_n) \rightarrow B$ die atomaren Sätze A_1, \dots, A_n bereits in \mathcal{T} enthalten ist, so füge auch B zu \mathcal{T} hinzu.
- ③ Wiederhole Schritt 2 so lange als möglich.
- ④ Falls $\perp \in \mathcal{T}$, ist S nicht WT-erfüllbar.

Andernfalls belege alle atomaren Sätze in \mathcal{T} mit wahr, und die anderen mit falsch. Die so erhaltene Belegung macht S wahr.

Theorem

Der Erfüllbarkeitsalgorithmus für Hornformeln ist korrekt, d. h. er klassifiziert genau die WT-erfüllbaren Hornformeln als WT-erfüllbar.

Korrektheit des Erfüllbarkeitsalgorithmus für Hornformeln

Beweis der Korrektheit des Algorithmus, Teil 1.

- Sei \mathcal{T}_1 die Menge der in Schritt 1 des Algorithmus in die Menge \mathcal{T} aufgenommenen atomaren Sätze.
- Sei \mathcal{T}_{n+1} die Menge der im n -ten Durchlauf von Schritt 2 in die Menge \mathcal{T} aufgenommenen Sätze.
- Sei $S \equiv S_1 \wedge \dots \wedge S_m$ die Hornformel, bestehend aus der Menge $Imp = \{S_1, \dots, S_m\}$ der Implikationen von S .
- Wir können dann $\mathcal{T}_n \subseteq \mathcal{A} \cup \{\perp, \top\}$ formal definieren als:
$$\mathcal{T}_0 = \{\top\}$$
$$\mathcal{T}_{n+1} = \mathcal{T}_n \cup \{B \mid (A_1 \wedge \dots \wedge A_k) \rightarrow B \in Imp, \{A_1, \dots, A_k\} \subseteq \mathcal{T}_n\}$$
- Da S endlich ist, gibt es ein N mit $\mathcal{T}_N = \mathcal{T}_{N+1} (= \mathcal{T})$.
- Dann gilt: $\perp \in \mathcal{T}_N$ gdw der Algorithmus sagt "unerfüllbar".



Korrektheit des Erfüllbarkeitsalgorithmus für Hornformeln

Beweis der Korrektheit des Algorithmus, Teil 2.

- ① Wenn S erfüllbar ist, dann $\perp \notin \mathcal{T}_N$
 - Sei h eine erfüllende Belegung für S , d.h. $\hat{h}(S) = T$.
 - Zeige per Induktion: $\hat{h}(A) = T$ für $A \in \mathcal{T}_N$.
 - Da $\hat{h}(\perp) = F$, muss $\perp \notin \mathcal{T}_N$ sein.
- ② Wenn $\perp \notin \mathcal{T}_N$, dann ist S erfüllbar
 - Wähle $h(A) = T$ gdw $A \in \mathcal{T}_N$



Aussagenlogisches Prolog

```
AncestorOf(X,Y) :- MotherOf(X,Y).  
AncestorOf(X,Y) :- FatherOf(X,Y).  
AncestorOf(X,Z) :- AncestorOf(X,Y),AncestorOf(Y,Z).  
MotherOf(mary,peter).  
FatherOf(peter,paul).  
FatherOf(peter,chris).
```

Um aussagenlogische atomare Sätze zu erhalten, müssen die Variablen X, Y und Z nacheinander durch alle möglichen Konstanten substituiert werden.

Aussagenlogisches Prolog

AncestorOf(mary,peter) :- MotherOf(mary,peter).

AncestorOf(peter,paul) :- MotherOf(peter,paul).

AncestorOf(mary,peter) :- FatherOf(mary,peter).

AncestorOf(peter,paul) :- FatherOf(peter,paul).

AncestorOf(mary,paul) :- AncestorOf(mary,peter),
AncestorOf(peter,paul).

MotherOf(mary,peter).

FatherOf(peter,paul).

FatherOf(peter,chris).

Für die Frage, ob diese logische Theorie zu B führt, fügt Prolog den Satz $\perp \leftarrow B$ hinzu und führt den Erfüllbarkeits-Algorithmus von Horn aus.

Falls der Algorithmus „unerfüllbar“ ergibt, antwortet Prolog mit „ja“, andernfalls mit „nein“

Aussagenlogisches Prolog in der Praxis

- Installieren Sie SWI-Prolog
- Laden Sie das Beispiel `ancestor.pl` herunter
- Geben Sie folgendes ein:

```
swipl  
?- [ancestor].  
true.  
?- ancestorOf(mary,paul).  
true
```

Zusammenfassung Veranstaltung 8

Zusammenfassung Veranstaltung 8 und Ausblick

Was haben wir heute gelernt?

- SAT-Solving und den DPLL-Algorithmus
- den Erfüllbarkeits-Algorithmus für Hornformeln

Nächstes Mal behandeln wir:

- dreiwertige Logik
 - gleiche Syntax wie bisher
 - aber andere Semantik, andere Begriffe von logischer Folgerung etc.

Lernziele Veranstaltung 9

Lernziele für heute (Veranstaltung 9)

- ich lerne **dreiwertige Logik** kennen
 - gleiche Syntax wie Aussagenlogik, aber andere Semantik
- ich vertiefe mein Verständnis des **Unterschieds von Syntax und Semantik**

Literatur:

Siegfried Gottwald. Mehrwertige Logik: eine Einführung in Theorie und Anwendungen, Akademie-Verlag, Berlin 1989.

Motivation

Zweiwertige Logik

- Aussagenlogik und PL1 sind **zweiwertig**
 - die Semantik von Sätzen ist ein Wahrheitswert in {T, F}
 - die Junktoren sind Funktionen über {T, F}
- Funktionen in PL1 (als Semantik von Funktionssymbolen) sind immer **total**

Warum dreiwertige Logik?

- In der Informatik sind Werte oft **undefiniert** bzw. Funktionen **partiell**:
 - in einer Datenbank sind nicht alle Felder eines Datensatzes mit Werten befüllt ("NULL-Einträge")
 - in einem Objekt wird versucht, ein Attribute auzulesen, das nicht initialisiert ist
 - Funktionen sind partiell, d.h. nicht überall definiert, z.B. $1/0$
- Verschiedene Umgangsmöglichkeiten mit undefinierten Werten in atomaren Formeln:
 - $P(1/0)$ falsch — bleibt in **zweiwertiger Logik**
 - $P(1/0)$ undefiniert — benötigt einen **dritten Wahrheitswert**

Beispiel: Object constraint Logik (OCL)

- OCL erlaubt, das **Verhalten von Objekten** zu spezifizieren
- Objektorientierter Code kann **verifiziert** werden, ob er eine OCL-Spezifikation erfüllt
- OCL ist standardisiert durch OMG, für die Unified modeling language UML und erbt UMLs **objektorientierte Notation**
- **Quantifizierung** über Objekte
- Beispiel einer **Invariante** (muss in allen Objektzuständen gelten):
-- Managers get a higher salary than employees
inv Branch2:
 self.employee->forall(e | e <> self.manager
 implies self.manager.salary > e.salary)
- es kann vorkommen, dass `e.salary` noch nicht initialisiert wurde
 - `self.manager.salary > e.salary` hat dann **Wahrheitswert U** (undefiniert)

Beispiel: Structured query language (SQL)

- Sprache zur Definition und Abfrage von Datenbanken
- Abfragesprache hat in etwa die Ausdrucksmächtigkeit von PL1
- Beispiel einer **Datenbanktabelle "Vorlesung"**:

Nr	Titel	Dozierende
5001	Einführung in die Informatik	Rössl
5002	Logik	Mossakowski
5003	Datenbanken	NULL

- es kann vorkommen, dass einzelne Felder nicht ausgefüllt sind (NULL)
- `SELECT Titel FROM Vorlesung WHERE Dozierende = 'Rössl'`
ist eine Abfrage, die nach allen Vorlesungen mit Dozierendem Rössl sucht
 - `NULL = 'Rössl'` hat dann **Wahrheitswert U** (undefiniert)

Kleene-Logik

Dreiwertige Logik von S.C. Kleene

- Logik von S.C. Kleene, um mit Undefiniertheiten umzugehen
- Wahrheitswerte = wahr (T), falsch (F), **undefiniert** (U)
- atomare Sätze haben Wahrheitswert U, wenn ein enthaltener Term undefiniert ist
- hier beschränken wir uns auf die Aussagenlogik

Wahrheitstafel für Konjunktion in der Kleene-Logik

\wedge	T	U	F
T	T		F
U			
F	F		F

Auf den bekannten Wahrheitswerten $\{T, F\}$ verhält sich Kleene-Logik wie klassische zweiwertige Logik
(man sagt auch, Konjunktion in Kleene-Logik ist **normal**)

Wahrheitstafel für Konjunktion in der Kleene-Logik

\wedge	T	U	F
T	T		F
U			F
F	F	F	F

Wenn eines der Argumente falsch ist, ist die ganze Konjunktion falsch, so wie in klassischer zweiwertiger Logik
(man sagt auch, Konjunktion in Kleene-Logik ist **uniform**)

Wahrheitstafel für Konjunktion in der Kleene-Logik

\wedge	T	U	F
T	T	U	F
U	U	U	F
F	F	F	F

Die restlichen Felder werden in Kleene-Logik mit U ausgefüllt.

Wahrheitstafeln der Kleene-Logik

Die anderen Wahrheitstafeln werden nach den gleichen Prinzipien aufgestellt:

\wedge	T	U	F
T	T	U	F
U	U	U	F
F	F	F	F

\vee	T	U	F
T	T	T	T
U	T	U	U
F	T	U	F

\neg	
T	F
U	U
F	T

\rightarrow	T	U	F
T	T	U	F
U	T	U	U
F	T	T	T

\leftrightarrow	T	U	F
T	T	U	F
U	U	U	U
F	F	U	T

\perp bedeutet immer F.

Wahrheitswertbelegung und Bewertungsfunktion in Kleene-Logik

Definition

Eine **Wahrheitswertbelegung** ist eine Funktion h von der Menge der atomaren Sätze einer Sprache in die Menge $\{T, F, U\}$.

Definition (Formalisierung der Wahrheitstafeln)

Die Wahrheitswertbelegung h kann zur **Bewertungsfunktion** \hat{h} von der Menge aller Sätze in die Menge $\{T, F, U\}$ **erweitert** werden. Dies geschieht gemäß der Wahrheitstafeln auf Folie 284.

Ausgezeichnete Wahrheitswerte und logische Wahrheiten

Definition

Die Menge der **ausgezeichneten Wahrheitswerte** in Kleene-Logik ist $D = \{T\}$. (D steht für “designated”.)

Wir erinnern uns:

Logisch notwendige Sätze oder **logische Wahrheiten** sind Sätze, die unter allen Umständen (in allen Welten) wahr sind.
“Wahr” wird nun durch “ausgezeichnet” ersetzt!

Definition

Ein Satz P ist Wahrheitstafel-wahr, WT-wahr oder eine **Tautologie**, wenn für alle Wahrheitswertbelegungen h gilt: $\hat{h}(P) \in D$.

Tautologien in Kleene-Logik

Durch Inspektion der Wahrheitstafeln erhält man folgenden Satz:

Theorem

Sei P ein \perp -freier Satz der Aussagenlogik. Wenn $h(A) = \text{U}$ für alle atomaren Sätze A , die in P vorkommen, dann ist auch $\hat{h}(P) = \text{U}$.

Korollar

In Kleene-Logik gibt es keine \perp -freien Tautologien.

Erfüllbarkeit

Definition

Eine logische Theorie \mathcal{T} ist **Wahrheitstafel-erfüllbar** (WT-erfüllbar) wenn für mindestens eine Wahrheitswertbelegung h gilt:

$$\hat{h}(P) \in D \text{ für alle } P \in \mathcal{T}.$$

Theorem

Eine logische Theorie ist in Kleene-Logik erfüllbar genau dann wenn sie es in zweiwertiger Aussagenlogik ist.

Logische Äquivalenz in Kleene-Logik

Wir übernehmen direkt die Definition aus der zweiwertigen Logik:

Definition

Zwei Sätze P und Q heißen **Kleene-tautologisch äquivalent** (in Zeichen $P \Leftrightarrow_{KL} Q$), wenn sie für alle Wahrheitswertbelegungen der atomaren Sätze den gleichen Wahrheitswert haben, d.h. für alle Wahrheitswertbelegungen h gilt: $\hat{h}(P) = \hat{h}(Q)$.

Um tautologische Äquivalenz zu prüfen, teste, ob P und Q in jeder Zeile der **Wahrheitstafel** den gleichen Wert erhalten.

Tautologische Äquivalenzen in Kleene-Logik

Es gelten die bekannten tautologischen Äquivalenzen aus der zweiwertigen Logik (Folien 128 und 130), bis auf:

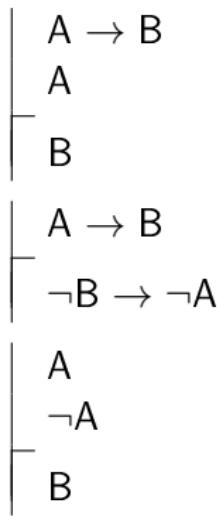
$$P \vee \neg P \not\vdash_{KI} \neg \perp \text{ Ausgeschlossenes Drittes}$$

Logische Folgerungen in Kleene-Logik

Definition

Ein Satz Q (Konklusion) ist eine **tautologische Folgerung** aus Prämissen P_1, \dots, P_n , wenn für alle Wahrheitsbelegungen h mit $\hat{h}(P_1) \in D$ und ... und $\hat{h}(P_n) \in D$ gilt: auch $\hat{h}(Q) \in D$.

Beispiele logischer Folgerungen in Kleene-Logik



Keine logische Folgerung in Kleene-Logik

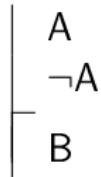
Folgende Argumente sind klassisch gültig, aber nicht in Kleene-Logik:

$$\begin{array}{l} \neg(P \leftrightarrow Q) \\ \quad \vdash (P \leftrightarrow R) \vee (Q \leftrightarrow R) \\ \\ \quad \vdash A \rightarrow A \\ \\ \quad \vdash A \vee \neg A \end{array}$$

Priest-Logik

Priest-Logik

- **Parakonsistenz:** für Datenbanken sind lokale Inkonsistenzen o.k. und führen nicht zu globaler Inkonsistenz
- wir wollen daher vermeiden, dass folgendes Argument gültig ist:



Definition

Die Priest-Logik ist definiert wie die Kleene-Logik, nur mit $D = \{T, U\}$ als Menge der **ausgezeichneten Wahrheitswerte**.

U spielt hier die Rolle von “sowohl wahr als auch falsch” und wird auch mit B (both) bezeichnet.

Tautologien und Erfüllbarkeit

Tautologien und Erfüllbarkeit sind genau wie in der Kleene-Logik definiert, nur dass wir ein anderes D verwenden.

Definition

Ein Satz P ist Wahrheitstafel-wahr, WT-wahr oder eine Tautologie, wenn für alle Wahrheitswertbelegungen h gilt: $\hat{h}(P) \in D$.

Definition

Eine logische Theorie \mathcal{T} ist Wahrheitstafel-erfüllbar (WT-erfüllbar) wenn für mindestens eine Wahrheitswertbelegung h gilt:

$$\hat{h}(P) \in D \text{ für alle } P \in \mathcal{T}.$$

Tautologien und Erfüllbarkeit in Priest-Logik

Tautologien und Erfüllbarkeit verhalten sich in Priest-Logik anders als in Kleene-Logik, da die Menge der ausgezeichneten Wahrheitswerte anders ist: $D = \{T, U\}$.

Theorem

Die Tautologien in Priest-Logik sind genau die Tautologien der klassischen zweiwertigen Logik.

Theorem

Jede \perp -freie logische Theorie ist in Priest-Logik erfüllbar.

Insbesondere ist die logische Theorie $\{A, \neg A\}$ erfüllbar (wichtig für parakonsistente Datenbanken).

Tautologische Äquivalenzen in Priest-Logik

Da die Wahrheitstafeln von Kleene-Logik und Priest-Logik übereinstimmen, sind auch die tautologischen Äquivalenzen die gleichen, das heißt:

Es gelten die bekannten tautologischen Äquivalenzen aus der zweiwertigen Logik (Folien 128 und 130), bis auf:

$$P \vee \neg P \not\Rightarrow_{KI} \neg \perp \text{ Ausgeschlossenes Drittes}$$

Logische Folgerung in Priest-Logik

Wir wiederholen die Definition:

Definition

Ein Satz Q (Konklusion) ist eine **tautologische Folgerung** aus Prämissen P_1, \dots, P_n , wenn für alle Wahrheitsbelegungen h mit $\hat{h}(P_1) \in D$ und ... und $\hat{h}(P_n) \in D$ gilt: auch $\hat{h}(Q) \in D$.

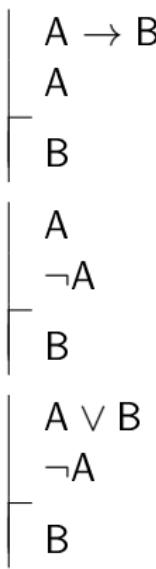
Der Begriff der logischen Folgerung ist in Priest-Logik anders als in Kleene-Logik, da die Menge der ausgezeichneten Wahrheitswerte anders ist: $D = \{\text{T}, \text{U}\}$.

Beispiele logischer Folgerungen in Priest-Logik

A \wedge B
|
A
|
A
|
A \vee B
|
A \rightarrow B
|
 $\neg B \rightarrow \neg A$

Keine logische Folgerung in Priest-Logik

Folgende Argumente sind klassisch gültig, aber nicht in Priest-Logik:



SAT-Solving

SAT-Solving

Theorem

Erfüllbarkeit, Tautologie und logische Folgerung sowohl in Kleene- als auch in Priest-Logik lassen sich als SAT-Problem in zweiwertiger Logik kodieren. Die Kodierung ist effizient, d.h. die Größe der Ausgabe lässt sich abschätzen durch ein Polynom über der Größe der Eingabe.

Beweisidee: kodiere Fakten wie $h(A) = \text{T}$, $h(A) = \text{F}$, $h(A) = \text{U}$ als je eigene Aussagenbuchstaben in zweiwertiger Logik. Dann lassen sich die Semantik von Kleene- als auch von Priest-Logik und darauf basierend auch Erfüllbarkeit und logische Folgerung als zweiwertiges Erfüllbarkeitsproblem kodieren.

Zusammenfassung Veranstaltung 9

Zusammenfassung Veranstaltung 9 und Ausblick

Was haben wir heute gelernt?

- **dreiwertige Logik**
 - gleiche Syntax wie klassische Aussagenlogik
 - andere Semantik
 - Logik für undefinierte Werte: Kleene-Logik
 - parakonsistente Logik: Priest-Logik
 - jeweils **anderer Folgerungsbegriff** als in zweiwertiger Logik
 - teils andere Tautologien (Kleene-Logik)
 - teils andere Erfüllbarkeiten (Priest-Logik)
 - Anwendungen

Nächstes Mal behandeln wir:

- Syntax und informelle Semantik von **Quantoren**
- Formalisierung mit Hilfe von Quantoren

Lernziele Veranstaltung 10

Lernziele für heute (Veranstaltung 10)

- ich lerne Syntax und informelle Semantik von Quantoren kennen.
- ich lerne, wie man deutsche Sätze mit Hilfe von Quantoren formalisiert
- ich lerne, wie man Argumente mit Hilfe von Quantoren formalisiert

Buch: 3.1–3.4, 9.1–9.6, 11.1–11.5

Syntax der PL1 mit Quantoren

Quantoren: Motivierende Beispiele

$\forall x \text{ } Cube(x)$ („Alle Objekte sind Würfel.“)

$\forall x (\text{ } Cube(x) \rightarrow Large(x))$ („Alle Würfel sind groß.“)

$\forall x \text{ } Large(x)$ („Alle Objekte sind groß.“)

$\exists x \text{ } Cube(x)$

„Es existiert ein Würfel.“

$\exists x (\text{ } Cube(x) \wedge Large(x))$

„Es existiert ein großer Würfel.“

Wiederholung: Sprache erster Stufe

Definition

Eine Sprache erster Stufe besteht aus

- einer Menge von Prädikatsymbolen mit Stelligkeiten, wie z. B. *Smaller*⁽²⁾, *Dodec*⁽¹⁾, *Between*⁽³⁾, \leq ⁽²⁾, einschließlich der aussagenlogischen Symbole (nullstellige Prädikatsymbole), wie z. B. $A^{(0)}$, $B^{(0)}$, $C^{(0)}$, (großgeschrieben),
 - das Gleichheitssymbol $=$ ist stets ein zweistelliges Prädikatsymbol,
- einer Menge von Namen oder Individuenkonstanten, wie z. B. a , b , c , (kleingeschrieben)
- einer Menge von Funktionssymbolen mit Stelligkeiten, wie z. B. $f^{(1)}$, $^{+(2)}$, $^{ \times (2)}$ (kleingeschrieben).

Üblicherweise werden die Stelligkeiten weggelassen.

Terme

Definition

Die Menge der **Terme** über einer Sprache erster Stufe ist wie folgt definiert:

$$\begin{array}{ll} t ::= c & \text{Individuenkonstanten} \\ | x & \text{Variablen (neu!)} \\ | f^{(n)}(t_1, \dots, t_n) & \begin{array}{l} \text{Anwendung von Funktionssymbolen} \\ \text{auf Terme } t_1, \dots, t_n \\ (\text{Rekursive Definition}) \end{array} \end{array}$$

Üblicherweise werden die Stelligkeiten weggelassen.

Variablen sind t, u, v, w, x, y, z , auch mit Indizes.

Individuenkonstanten sind a, b, c, d, e, f, n , und andere.

Wohlgeformte Formeln

Definition

Die Menge der **wohlgeformten Formeln** über einer Sprache erster Stufe ist wie folgt induktiv definiert:

$P ::= Pr(t_1, \dots, t_n)$	Anwendung von Prädikatsymbolen auf Terme
\perp	Widerspruch
$\neg P$	Negation
$(P_1 \wedge P_2)$	Konjunktion
$(P_1 \vee P_2)$	Disjunktion
$(P_1 \rightarrow P_2)$	Implikation, Konditional
$(P_1 \leftrightarrow P_2)$	Biimplikation, Bikonditional
$\forall x P$	Allquantor
$\exists x P$	Existenzquantor

$= (t_1, t_2)$ wird als $t_1 = t_2$ notiert.

Aufbau von $\exists y (Mag(a, y) \wedge \neg Mag(y, y))$

$\exists y (Mag(a, y) \wedge \neg Mag(y, y))$



$Mag(a, y) \wedge \neg Mag(y, y)$



$Mag(a, y)$

$\neg Mag(y, y)$



$Mag(y, y)$

Klammern

Die äußeren Klammern einer wohlgeformten Formel können weggelassen werden.

$$\textit{Cube}(x) \wedge \textit{Small}(x)$$

Im Allgemeinen sind Klammern wichtig, um den Einflussbereich (*Skopus*) von Quantoren festzulegen.

Freie und gebundene Variablen

Die Vorkommen von x in $\forall x P(x)$ und $\exists x P(x)$ nennt man **gebunden**.

Ein Vorkommen einer Variable in einer wohlgeformten Formel, die nicht durch einen Quantor gebunden ist, nennt man **frei**.

$\exists y \text{ LeftOf}(x, y)$	x ist frei, y ist gebunden
$(\text{Cube}(x) \wedge \text{Small}(x))$ $\rightarrow \exists y \text{ LeftOf}(x, y)$	x ist frei, y ist gebunden
$\exists x (\text{Cube}(x) \wedge \text{Small}(x))$	Beide Vorkommen von x sind gebunden
$\exists x \text{ Cube}(x) \wedge \text{Small}(x)$	Das erste Vorkommen von x ist gebunden, das zweite ist frei

Sätze

Definition

Ein **Satz** ist eine wohlgeformte Formel ohne freie Variable.

\perp

$A \wedge B$

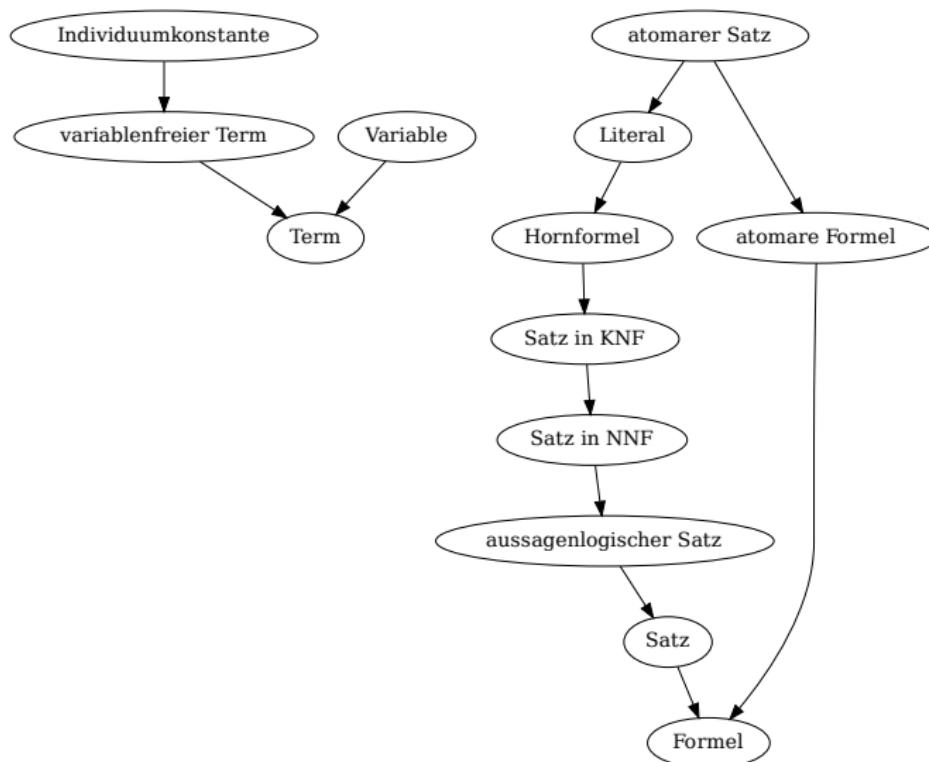
$\text{Cube}(a) \vee \text{Tet}(b)$

$\forall x (\text{Cube}(x) \rightarrow \text{Large}(x))$

$\forall x ((\text{Cube}(x) \wedge \text{Small}(x)) \rightarrow \exists y \text{ LeftOf}(x, y))$

Bemerkung: Alle aussagenlogischen Sätze im Sinne von Folie 59 sind auch Sätze im Sinne obigen Sinne. Insofern ist die Bezeichnung "Satz" hier angemessen.

Hierarchie der Begriffe



Informelle Semantik der PL1 mit Quantoren

Die Semantik der Quantoren

- Die Aussagen quantifizierter Sätze beziehen sich auf einen nichtleeren intendierten **Gegenstandsbereich** (domain of discourse).
- Ein Satz der Form $\forall x S(x)$ ist genau dann wahr, wenn die wohlgeformte Formel $S(n)$ für **jedes** Element n aus dem Gegenstandsbereich wahr ist.
- Ein Satz der Form $\exists x S(x)$ ist genau dann wahr, wenn die wohlgeformte Formel $S(n)$ für mindestens **ein** Element n aus dem Gegenstandsbereich wahr ist.
- Nicht alle Elemente aus dem Gegenstandsbereich müssen Namen haben – bei Bedarf können Namen n_1, n_2, \dots vergeben werden.

Das Henkin-Hintikka Spiel

© Original Artist
Reproduction rights obtainable from
www.CartoonStock.com



"Checkmate!"

Das Henkin-Hintikka-Spiel

Ist ein Satz in einer gegebenen Welt wahr?

- Spieler: Sie und der Computer (Bivalence World)
- Sie behaupten, ein Satz ist wahr (oder falsch), Bivalence World wird das Gegenteil behaupten.
- In jeder Runde wird der Satz reduziert und es entsteht ein einfacherer.
- Wenn ein atomarer Satz erreicht ist, kann sein Wahrheitswert direkt in der gegebenen Welt überprüft werden.

Sie haben eine Gewinnstrategie genau dann, wenn Ihre Behauptung wahr ist.

Henkin-Hintikka-Spiel: Die Spielregeln für Junktoren

Form	Ihre Entscheidung	Spieler am Zug	Ziel
$\neg P$	in beiden Fällen	—	Ersetze $\neg P$ durch P und kehre die Entscheidung um.
$P \wedge Q$	wahr	Bivalence World	Wähle eines von P, Q , das falsch ist.
	falsch	Sie	
$P \vee Q$	wahr	Sie	Wähle eines von P, Q , das wahr ist.
	falsch	Bivalence World	

$P \rightarrow Q$ wird durch $\neg P \vee Q$ ersetzt.

$P \leftrightarrow Q$ wird durch $(P \rightarrow Q) \wedge (Q \rightarrow P)$ ersetzt.

Henkin-Hintikka-Spiel: Die Spielregeln für Quantoren

Form	Ihre Festlegung	Spieler am Zug	Ziel
$\exists x P(x)$	wahr falsch	Sie Bivalence World	Ein b wählen, das die Wff $P(x)$ erfüllt.
$\forall x P(x)$	wahr falsch	Bivalence World Sie	Ein b wählen, das die Wff $P(x)$ nicht erfüllt.

Formalisierung in PL1 mit Quantoren

Die vier Aristotelischen Formen

Alle Ps sind Qs.

$$\forall x(P(x) \rightarrow Q(x))$$

Manche Ps sind Qs.

$$\exists x(P(x) \wedge Q(x))$$

Kein P ist ein Q.

$$\forall x(P(x) \rightarrow \neg Q(x))$$

Manche Ps sind keine Qs.

$$\exists x(P(x) \wedge \neg Q(x))$$

Bemerkung:

$\forall x(P(x) \rightarrow Q(x))$ bedeutet nicht, dass Ps existieren.

$\exists x(P(x) \wedge Q(x))$ bedeutet nicht, dass nicht alle Ps auch Qs sind.

Übersetzungsbeispiele

Es gibt einen kleinen Würfel.

$$\exists x (\text{Cube}(x) \wedge \text{Small}(x))$$

Würfel sind groß.

$$\forall x (\text{Cube}(x) \rightarrow \text{Large}(x))$$

Übersetzung komplexer Nominalphrasen

Ein kleiner, glücklicher Hund ist zu Hause.

$$\exists x \left((\text{Klein}(x) \wedge \text{Glücklich}(x) \wedge \text{Hund}(x)) \wedge \text{Zuhause}(x) \right)$$

Jeder kleine Hund, der zu Hause ist, ist glücklich.

$$\forall x \left((\text{Klein}(x) \wedge \text{Hund}(x) \wedge \text{Zuhause}(x)) \rightarrow \text{Glücklich}(x) \right)$$

Leererweise wahre Aussagen

$$\forall y (\text{Tet}(y) \rightarrow \text{Small}(y))$$

gilt auch in Welten ohne Tetraeder,

$$\forall y (\text{Tet}(y) \rightarrow \text{Cube}(y))$$

gilt nur in Welten ohne Tetraeder.

Mehrfache Quantoren in Aristotelischen Formen

Ein Würfel befindet sich links von einem Tetraeder.

$$\exists x \exists y ((\text{Cube}(x) \wedge \text{Tet}(y)) \wedge \text{LeftOf}(x, y))$$

$$\exists x ((\text{Cube}(x) \wedge \exists y (\text{Tet}(y) \wedge \text{LeftOf}(x, y)))$$

Jeder Würfel befindet sich links von allen Tetraedern.

$$\forall x \forall y ((\text{Cube}(x) \wedge \text{Tet}(y)) \rightarrow \text{LeftOf}(x, y))$$

$$\forall x ((\text{Cube}(x) \rightarrow \forall y (\text{Tet}(y) \rightarrow \text{LeftOf}(x, y)))$$

Mehrfache Quantoren und konversationale Implikatur

Was besagt (in Bivalence World) der Satz

$$\forall x \forall y ((\text{Cube}(x) \wedge \text{Cube}(y)) \rightarrow (\text{LeftOf}(x, y) \vee \text{RightOf}(x, y))) ?$$

Was besagt der Satz

$$\exists x \exists y (\text{Cube}(x) \wedge \text{Cube}(y)) ?$$

Gemischte Quantoren

Jeder Würfel befindet sich links von einem Tetraeder.

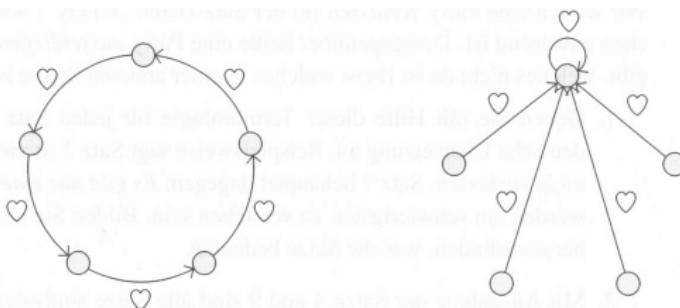
$$\forall x \left(\text{Cube}(x) \rightarrow \exists y (\text{Tet}(y) \wedge \text{LeftOf}(x, y)) \right)$$

Die Reihenfolge gemischter Quantoren

$$\forall x \exists y \text{ Mag}(x, y)$$

ist **nicht logisch äquivalent zu**

$$\exists y \forall x \text{ Mag}(x, y)$$



Eine Situation, in der $\forall x \exists y \text{ Mag}(x, y)$ wahr ist,
im Vergleich mit einer Situation, in der $\exists y \forall x \text{ Mag}(x, y)$ gilt.

Es gibt genau ein . . .

Es gibt genau einen Würfel.

$$\exists x (Cube(x) \wedge \forall y (Cube(y) \rightarrow y = x))$$

Übersetzen mit der Schritt-für-Schritt-Methode

Jeder Würfel befindet sich links von einem Tetraeder.



$$\forall x (Cube(x) \rightarrow x \text{ befindet-sich-links-von-einem-Tetraeder})$$

$$x \text{ befindet-sich-links-von-einem-Tetraeder} \rightsquigarrow \exists y (Tet(y) \wedge LeftOf(x, y))$$

$$\forall x (Cube(x) \rightarrow x \text{ befindet-sich-links-von-einem-Tetraeder})$$



$$\forall x (Cube(x) \rightarrow \exists y (Tet(y) \wedge LeftOf(x, y)))$$

Übersetzen mit der Schritt-für-Schritt-Methode II

Hinter einem Würfel befindet sich ein Tetraeder.

~→

$$\exists x (Cube(x) \wedge \text{ein-Tetraeder-befindet-sich-hinter } x)$$

ein-Tetraeder-befindet-sich-hinter x ~→ $\exists y (Tet(y) \wedge BackOf(y, x))$

$$\exists x (Cube(x) \wedge \text{ein-Tetraeder-befindet-sich-hinter } x)$$

~→

$$\exists x (Cube(x) \wedge \exists y (Tet(y) \wedge BackOf(y, x)))$$

Paraphrasierung ist manchmal notwendig

Jeder Bauer, der einen Esel besitzt, füttert ihn.

$\forall x (Bauer(x) \wedge \exists y (Esel(y) \wedge Besitzt(x, y)) \rightarrow Füttert(x, y))$ falsch!

Paraphrasieren:

Jeder Esel wird von jedem Bauern, der ihn besitzt, gefüttert.

$\forall x (Esel(x) \rightarrow \forall y ((Bauer(y) \wedge Besitzt(y, x)) \rightarrow Füttert(y, x)))$

Mehrdeutigkeit und Kontextabhängigkeit

Jede Minute wird in New York ein Mann überfallen.
Wir werden ihn heute Abend interviewen.

Schwache Lesart:

$$\forall x (\text{Minute}(x) \rightarrow \exists y (\text{Mann}(y) \wedge \text{ÜberfallenWährend}(y, x)))$$

Starke Lesart:

$$\exists y (\text{Mann}(y) \wedge \forall x (\text{Minute}(x) \rightarrow \text{ÜberfallenWährend}(y, x)))$$

Formalisierung von Argumenten in PL1 mit Quantoren

Formalisierung von Argumenten

Niemand verlässt den Raum. Also auch Peter nicht.

 | Niemand verlässt den Raum.

 | Peter verlässt nicht den Raum.

 | $\neg \exists x \text{ VerlässtRaum}(x)$

 | $\neg \text{VerlässtRaum}(\text{peter})$

Ist dieses Argument gültig oder nicht? Warum?

Formalisierung von Argumenten II

a ist ein Würfel. Schließlich ist a groß, und alle Würfel sind groß.

a ist groß.

Alle Würfel sind groß.

a ist ein Würfel.

$\text{Large}(a)$

$\forall x (\text{Cube}(x) \rightarrow \text{Large}(x))$

$\text{Cube}(a)$

Ist dieses Argument gültig oder nicht? Warum?

Zusammenfassung

Veranstaltung 10

Zusammenfassung Veranstaltung 10 und Ausblick

Was haben wir heute gelernt?

- Syntax und Semantik von Quantoren
 - Formalisierung mit Quantoren

Nächstes Mal behandeln wir:

- PL1-Strukturen
 - Tarski's Wahrheitsbegriff
 - Unterschied zwischen Tautologie und PL1-Wahrheit
 - Unterschied zwischen tautologischer und PL1-Folgerung

Lernziele Veranstaltung 11

Lernziele für heute (Veranstaltung 11)

- ich lerne eine präzise **Semantik von PL1** kennen:
PL1-Strukturen
- ich lerne **Tarski's Wahrheitsbegriff** kennen
(heute ohne Quantoren, die nächstes Mal dran kommen)
- ich lerne, wie man **PL1-Wahrheiten** erkennt
und von anderen Wahrheiten abgrenzt
- ich lerne, wie man **PL1-Folgerungen** erkennt
und von anderen Folgerungen abgrenzt

Buch: 18.1, 18.2 (Band 2)

PL1-Strukturen

Wiederholung: Die (informelle) Semantik der Quantoren

- Die Aussagen quantifizierter Sätze beziehen sich auf einen nichtleeren intendierten **Gegenstandsbereich** (domain of discourse).
- Ein Satz der Form $\forall x S(x)$ ist genau dann wahr, wenn die wohlgeformte Formel $S(n)$ für **jedes** Element n aus dem Gegenstandsbereich wahr ist.
- Ein Satz der Form $\exists x S(x)$ ist genau dann wahr, wenn die wohlgeformte Formel $S(n)$ für mindestens **ein** Element n aus dem Gegenstandsbereich wahr ist.
- Nicht alle Elemente aus dem Gegenstandsbereich müssen Namen haben – bei Bedarf können Namen n_1, n_2, \dots vergeben werden.

PL1-Strukturen – Motivation

- Wie können wir die **Semantik der Quantoren** und den Begriff der **logischen Folgerung** formaler fassen?
- In der Aussagenlogik führte die Wahrheitstafelmethode zu einer Präzisierung des Begriffs „logische Folgerung“, nämlich zum Begriff der **tautologischen Folgerung**.
- In der Prädikatenlogik erster Stufe müssen wir aber auch die Quantoren (\forall, \exists) sowie die Identität (=) interpretieren.
- Der Begriff der **PL1-Struktur** (Struktur der Prädikatenlogik erster Stufe) modelliert Situationen von Bivalence World und auch Situationen der realen Welt durch Verwendung der **Mengentheorie**.

PL1-Strukturen: (vorläufige) Definition

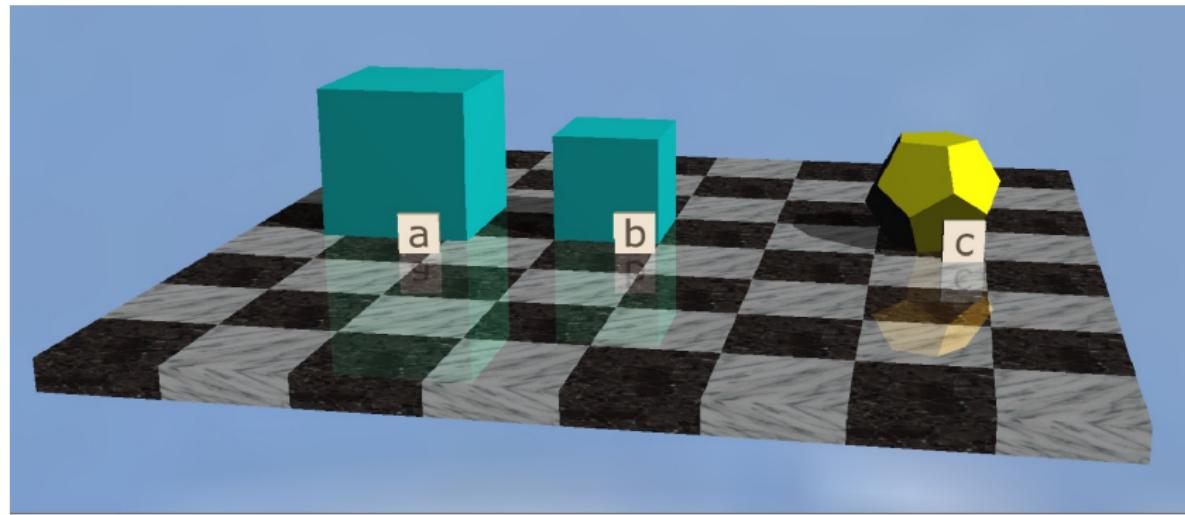
Eine PL1-Struktur \mathfrak{M} enthält

- eine nichtleere Menge $D^{\mathfrak{M}}$, dem **Gegenstandsbereich**,
- für jeden Namen (Individuenkonstante) c der Sprache ein Element $\mathfrak{M}(c)$ aus $D^{\mathfrak{M}}$,
- für jedes n -stellige Prädikatsymbol Pr der Sprache eine Menge $\mathfrak{M}(Pr) \subseteq \underbrace{D^{\mathfrak{M}} \times \cdots \times D^{\mathfrak{M}}}_{n \text{ mal}}$ von n -Tupeln (x_1, \dots, x_n) von Elementen aus $D^{\mathfrak{M}}$, auch **Extension** von Pr bezeichnet,
 - die Extension des Identitätssymbols „=“ wird festgelegt als $\{(x, x) \mid x \in D^{\mathfrak{M}}\}$.

$\mathfrak{M}(c)$ heißt auch die **Interpretation** von c in \mathfrak{M} .

$\mathfrak{M}(Pr)$ heißt auch die **Interpretation** von Pr in \mathfrak{M} .

Beispiel



Eine PL1-Struktur \mathfrak{M} entsprechend Bivalence World

Für eine PL1-Sprache bestehend aus den Prädikatsymbolen *Cube*, *Dodec* und *Larger* und den Konstanten *a*, *b* und *c* definieren wir

- $D^{\mathfrak{M}} = \{\begin{array}{c} \text{cyan cube} \\ \text{yellow cube} \end{array}, \text{dodecahedron}\},$
- $\mathfrak{M}(\text{Cube}) = \{\begin{array}{c} \text{cyan cube} \\ \text{yellow cube} \end{array}\},$
- $\mathfrak{M}(\text{Dodec}) = \{\text{dodecahedron}\},$
- $\mathfrak{M}(\text{Larger}) = \{(\begin{array}{c} \text{cyan cube} \\ \text{yellow cube} \end{array}), (\begin{array}{c} \text{cyan cube} \\ \text{dodecahedron} \end{array})\},$
- $\mathfrak{M}(=) = \{(\begin{array}{c} \text{cyan cube} \\ \text{cyan cube} \end{array}), (\begin{array}{c} \text{yellow cube} \\ \text{yellow cube} \end{array}), (\text{dodecahedron}, \text{dodecahedron})\},$
- $\mathfrak{M}(a) = \text{cyan cube}, \quad \mathfrak{M}(b) = \text{yellow cube}, \quad \mathfrak{M}(c) = \text{dodecahedron}.$

Verwenden Sie auch [Bivalence World](#), um PL1-Strukturen für Klötzenwelten darzustellen.

Eine PL1-Struktur \mathfrak{M}' , die nicht Bivalence World entspricht

- $D^{\mathfrak{M}'} = \{\text{Cube}, \text{Square}, \text{Dodec}\};$
- $\mathfrak{M}'(\text{Cube}) = \{\text{Dodec}, \text{Cube}\};$
- $\mathfrak{M}'(\text{Dodec}) = \emptyset;$
- $\mathfrak{M}'(\text{Larger}) = \{(\text{Cube}, \text{Cube}), (\text{Dodec}, \text{Cube})\};$
- $\mathfrak{M}'(=) = \{(\text{Cube}, \text{Cube}), (\text{Cube}, \text{Cube}), (\text{Dodec}, \text{Dodec})\};$
- $\mathfrak{M}'(a) = \text{Cube}; \quad \mathfrak{M}'(b) = \text{Square}; \quad \mathfrak{M}'(c) = \text{Dodec}.$

Der Wahrheitsbegriff von Alfred Tarski

Mit $\mathfrak{M} \models P$ wollen wir ausdrücken, dass der Satz P in der Struktur \mathfrak{M} **wahr** (oder **erfüllt**) ist. Wir definieren das rekursiv durch:

- ① $\mathfrak{M} \models Pr(c_1, \dots, c_n)$ g.d.w. $(\mathfrak{M}(c_1), \dots, \mathfrak{M}(c_n)) \in \mathfrak{M}(Pr)$,
 - wir wissen aus der Definition einer PL1-Struktur, dass $\mathfrak{M}(Pr)$ (die **Extension** von Pr) eine Menge von n -Tupeln (x_1, \dots, x_n) von Elementen aus $D^{\mathfrak{M}}$ ist,
- ② $\mathfrak{M} \not\models \perp$,
- ③ $\mathfrak{M} \models \neg P$ g.d.w. $\mathfrak{M} \not\models P$,
- ④ $\mathfrak{M} \models P \wedge Q$ g.d.w. $\mathfrak{M} \models P$ und $\mathfrak{M} \models Q$,
- ⑤ $\mathfrak{M} \models P \vee Q$ g.d.w. $\mathfrak{M} \models P$ oder $\mathfrak{M} \models Q$ oder beide,
- ⑥ $\mathfrak{M} \models P \rightarrow Q$ g.d.w. $\mathfrak{M} \not\models P$ oder $\mathfrak{M} \models Q$ oder beide,
- ⑦ $\mathfrak{M} \models P \leftrightarrow Q$ g.d.w. $(\mathfrak{M} \models P \text{ g.d.w. } \mathfrak{M} \models Q)$,

Dabei bedeutet $\mathfrak{M} \not\models P$, dass $\mathfrak{M} \models P$ nicht gilt, d.h. dass der Satz P in der Struktur \mathfrak{M} **falsch** ist.

Beispiel

$$D^{\mathfrak{M}} = \{anna, berta, cyrel\}$$

$$\mathfrak{M}(Mag) = \{(anna, anna), (anna, berta), (cyrel, anna)\}$$

$$\mathfrak{M}(a) = anna$$

$$\mathfrak{M}(b) = berta$$

$$\mathfrak{M} \models^? \neg a = b$$

Dies gilt, weil $(\mathfrak{M}(a), \mathfrak{M}(b)) = (anna, berta) \notin \mathfrak{M}(M =)$.

Beispiel

$$D^{\mathfrak{M}} = \{ anna, berta, cyrel \}$$

$$\mathfrak{M}(Mag) = \{ (anna, anna), (anna, berta), (cyrel, anna) \}$$

$$\mathfrak{M}(a) = anna$$

$$\mathfrak{M}(b) = berta$$

$$\mathfrak{M} \stackrel{?}{\models} Mag(a, b) \wedge \neg Mag(b, b)$$

Dies gilt, weil $(\mathfrak{M}(a), \mathfrak{M}(b)) = (anna, berta) \in \mathfrak{M}(Mag)$
 und $(\mathfrak{M}(b), \mathfrak{M}(b)) = (bertha, berta) \notin \mathfrak{M}(Mag)$.

PL1-Strukturen mit Funktionen

PL1-Strukturen: Definition (mit Funktionen)

Definition

Eine PL1-Struktur \mathfrak{M} enthält

- eine nichtleere Menge $D^{\mathfrak{M}}$, dem **Gegenstandsbereich**,
- für jeden Namen (Individuenkonstante) c der Sprache ein **Element** $\mathfrak{M}(c)$ aus $D^{\mathfrak{M}}$,
- für jedes n -stellige Prädikatsymbol Pr der Sprache eine Menge $\mathfrak{M}(\text{Pr})$ von n -Tupeln (x_1, \dots, x_n) von Elementen aus $D^{\mathfrak{M}}$, auch **Extension** von Pr bezeichnet,
 - die Extension des Identitätssymbols „=“ wird festgelegt als $\{(x, x) \mid x \in D^{\mathfrak{M}}\}$,
- **für jedes n -stellige Funktionensymbol f der Sprache eine n -stellige Funktion** $\mathfrak{M}(f): (D^{\mathfrak{M}})^n \rightarrow D^{\mathfrak{M}}$.

Beispiele für PL1-Strukturen mit Funktionen

- Die **natürlichen Zahlen** mit den Konstanten 0 und 1, zweistelligen Funktionen + und *, und dem zweistelligen Prädikat \leq
- Ähnlich: die **reellen Zahlen**
- Jede **Gruppe** ist eine PL1-Struktur. Ebenso jeder Ring, Körper, Vektorraum etc.
- Die Menge der **Wörter (Strings)** über dem Unicode-Alphabet, mit der Konstanten ε (leeres Wort), der zweistelligen Funktion \cdot (Hintereinanderschreiben von Wörtern) und dem zweistelligen Prädikat \leq (Teilwort)
- Die Menge der **binären Bäume** ohne Blätter, mit der Konstanten ε (leerer Baum), der zweistelligen Funktion bin (Zusammenfügen zweier Bäume) und dem zweistelligen Prädikat $IstKindVon$

Die Interpretation von Termen $\mathfrak{M}(t)$

Definition

Es seien \mathfrak{M} eine PL1-Struktur und t ein Term ohne Variablen.

Für t definieren wir $\mathfrak{M}(t)$ rekursiv durch

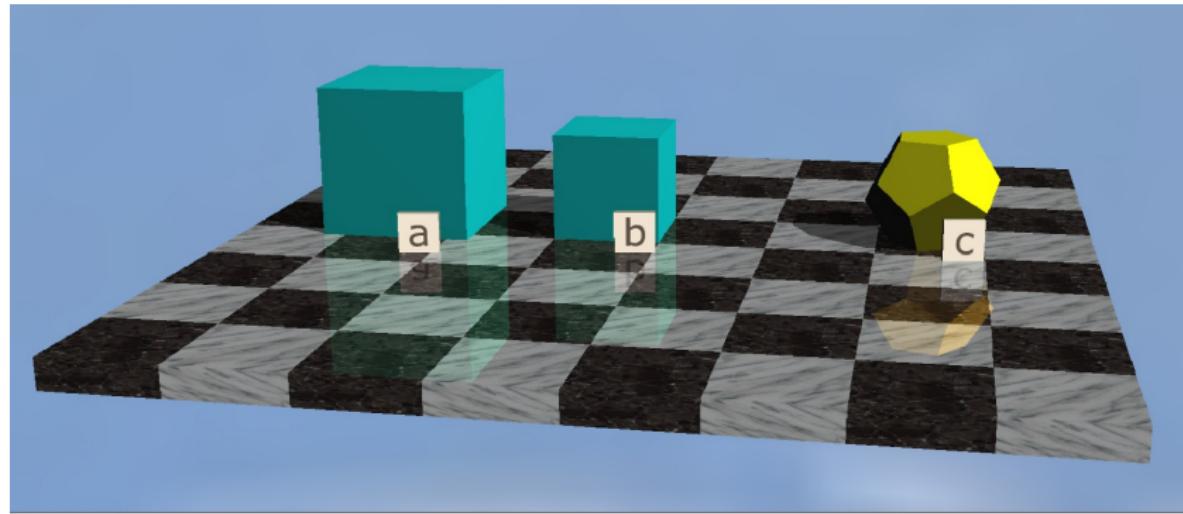
- $\mathfrak{M}(c)$, falls t eine Individuenkonstante c ist,
 - da t eine Individuenkonstante c ist, wissen wir aus der Definition einer PL1-Struktur, dass $\mathfrak{M}(c)$ ein Element aus $D^{\mathfrak{M}}$ ist,
- $\mathfrak{M}(f)(\mathfrak{M}(t_1), \dots, \mathfrak{M}(t_n))$, falls $t = f(t_1, \dots, t_n)$ ist,
 - wir wissen aus der Definition einer PL1-Struktur, dass $\mathfrak{M}(f)$ eine n -stellige Funktion $\mathfrak{M}(f): (D^{\mathfrak{M}})^n \rightarrow D^{\mathfrak{M}}$ ist.

Der Wahrheitsbegriff von Alfred Tarski (mit Funktionssymbolen)

Definition ($\mathfrak{M} \models P$ (Satz P in der Struktur \mathfrak{M} **wahr** oder **erfüllt**))

- ① $\mathfrak{M} \models Pr(t_1, \dots, t_n)$ g.d.w. $(\mathfrak{M}(t_1), \dots, \mathfrak{M}(t_n)) \in \mathfrak{M}(Pr)$,
- ② $\mathfrak{M} \not\models \perp$,
- ③ $\mathfrak{M} \models \neg P$ g.d.w. $\mathfrak{M} \not\models P$,
- ④ $\mathfrak{M} \models P \wedge Q$ g.d.w. $\mathfrak{M} \models P$ und $\mathfrak{M} \models Q$,
- ⑤ $\mathfrak{M} \models P \vee Q$ g.d.w. $\mathfrak{M} \models P$ oder $\mathfrak{M} \models Q$ oder beide,
- ⑥ $\mathfrak{M} \models P \rightarrow Q$ g.d.w. $\mathfrak{M} \not\models P$ oder $\mathfrak{M} \models Q$ oder beide,
- ⑦ $\mathfrak{M} \models P \leftrightarrow Q$ g.d.w. $(\mathfrak{M} \models P$ g.d.w. $\mathfrak{M} \models Q)$,

Beispiel



PL1-Struktur mit Funktionen

- $D^{\mathfrak{M}} = \{\begin{array}{c} \text{Cub} \\ \text{Cub} \\ \text{Dodec} \end{array}\},$
- $\mathfrak{M}(Cube) = \{\begin{array}{c} \text{Cub} \\ \text{Cub} \end{array}\},$
- $\mathfrak{M}(Dodec) = \{\begin{array}{c} \text{Dodec} \end{array}\},$
- $\mathfrak{M}(Larger) = \{(\begin{array}{c} \text{Cub} \\ \text{Cub} \end{array}), (\begin{array}{c} \text{Cub} \\ \text{Dodec} \end{array})\},$
- $\mathfrak{M}(=) = \{(\begin{array}{c} \text{Cub} \\ \text{Cub} \end{array}), (\begin{array}{c} \text{Cub} \\ \text{Cub} \end{array}), (\begin{array}{c} \text{Dodec} \\ \text{Dodec} \end{array})\},$
- $\mathfrak{M}(a) = \begin{array}{c} \text{Cub} \end{array}, \quad \mathfrak{M}(b) = \begin{array}{c} \text{Cub} \end{array}, \quad \mathfrak{M}(c) = \begin{array}{c} \text{Dodec} \end{array},$
- $\mathfrak{M}(Im) = \{\begin{array}{c} \text{Cub} \\ \text{Cub} \end{array} \mapsto \begin{array}{c} \text{Cub} \\ \text{Cub} \end{array}, \begin{array}{c} \text{Cub} \\ \text{Dodec} \end{array} \mapsto \begin{array}{c} \text{Cub} \\ \text{Cub} \end{array}, \begin{array}{c} \text{Dodec} \\ \text{Dodec} \end{array} \mapsto \begin{array}{c} \text{Cub} \\ \text{Cub} \end{array}\},$
- $\mathfrak{M}(rm) = \{\begin{array}{c} \text{Cub} \\ \text{Cub} \end{array} \mapsto \begin{array}{c} \text{Dodec} \\ \text{Dodec} \end{array}, \begin{array}{c} \text{Cub} \\ \text{Dodec} \end{array} \mapsto \begin{array}{c} \text{Dodec} \\ \text{Dodec} \end{array}, \begin{array}{c} \text{Dodec} \\ \text{Dodec} \end{array} \mapsto \begin{array}{c} \text{Dodec} \\ \text{Dodec} \end{array}\},$
- $\mathfrak{M}(fm) = \{\begin{array}{c} \text{Cub} \\ \text{Cub} \end{array} \mapsto \begin{array}{c} \text{Cub} \\ \text{Cub} \end{array}, \begin{array}{c} \text{Cub} \\ \text{Dodec} \end{array} \mapsto \begin{array}{c} \text{Cub} \\ \text{Cub} \end{array}, \begin{array}{c} \text{Dodec} \\ \text{Dodec} \end{array} \mapsto \begin{array}{c} \text{Dodec} \\ \text{Dodec} \end{array}\},$
- $\mathfrak{M}(bm) = \{\begin{array}{c} \text{Cub} \\ \text{Cub} \end{array} \mapsto \begin{array}{c} \text{Cub} \\ \text{Cub} \end{array}, \begin{array}{c} \text{Cub} \\ \text{Dodec} \end{array} \mapsto \begin{array}{c} \text{Cub} \\ \text{Cub} \end{array}, \begin{array}{c} \text{Dodec} \\ \text{Dodec} \end{array} \mapsto \begin{array}{c} \text{Dodec} \\ \text{Dodec} \end{array}\}.$

Beispiel mit Funktionssymbolen

$$\mathfrak{M} \stackrel{?}{\models} rm(a) = rm(c)$$

$$\mathfrak{M} \models rm(a) = rm(c)$$

g.d.w. $(\mathfrak{M}(rm(a)), \mathfrak{M}(rm(c))) \in \mathfrak{M}(=)$

g.d.w. $(\mathfrak{M}(rm)(\mathfrak{M}(a)), \mathfrak{M}(rm)(\mathfrak{M}(c))) \in \mathfrak{M}(=)$

g.d.w. $(\mathfrak{M}(rm)(\text{ \fbox{teal}}), \mathfrak{M}(rm)(\text{ \fbox{yellow}})) \in \mathfrak{M}(=)$

g.d.w. $(\text{ \fbox{yellow}}, \text{ \fbox{yellow}}) \in \mathfrak{M}(=)$

g.d.w. $(\text{ \fbox{yellow}}, \text{ \fbox{yellow}}) \in \{(\text{ \fbox{teal}}, \text{ \fbox{teal}}), (\text{ \fbox{teal}}, \text{ \fbox{teal}}), (\text{ \fbox{yellow}}, \text{ \fbox{yellow}})\}$

Letzteres ist der Fall, daher ist der Satz in \mathfrak{M} wahr.

Beispiel mit Funktionssymbolen II

$$\mathfrak{M} \models^? \text{Larger}(rm(a), a)$$

$$\mathfrak{M} \models \text{Larger}(rm(a), a)$$

g.d.w. $(\mathfrak{M}(rm(a)), \mathfrak{M}(a)) \in \mathfrak{M}(\text{Larger})$

g.d.w. $(\mathfrak{M}(rm)(\mathfrak{M}(a)), \mathfrak{M}(a)) \in \mathfrak{M}(\text{Larger})$

g.d.w. $(\mathfrak{M}(rm)(\text{█}), \text{█}) \in \mathfrak{M}(\text{Larger})$

g.d.w. $(\text{█}, \text{█}) \in \mathfrak{M}(\text{Larger})$

g.d.w. $(\text{█}, \text{█}) \in \{(\text{█}, \text{█}), (\text{█}, \text{█})\}$

Letzteres ist jedoch nicht der Fall, daher ist der Satz in \mathfrak{M} falsch.

PL1-Erfüllbarkeit und PL1-Wahrheit

PL1-Erfüllbarkeit

Definition

Ein Satz P heißt **PL1-erfüllbar** genau dann, wenn es eine PL1-Struktur gibt, die P erfüllt. Ansonsten heißt er **PL1-unerfüllbar**. Entsprechend heißt eine logische Theorie \mathcal{T} PL1-erfüllbar genau dann, wenn es eine PL1-Struktur gibt, die alle Sätze aus \mathcal{T} erfüllt.

PL1-Erfüllbarkeit: Beispiel

Ist $\{Large(a), Cube(a)\}$ PL1-erfüllbar?

PL1-Erfüllbarkeit: Beispiel II

Ist $\{Tet(a), Cube(a)\}$ PL1-erfüllbar?

PL1-Erfüllbarkeit: Beispiel III

Ist $\neg a = a$ PL1-erfüllbar?

PL1-Wahrheit

Definition

Ein Satz P heißt eine **PL1-Wahrheit** genau dann, wenn jede PL1-Struktur den Satz P erfüllt.

Theorem

*Ein Satz P ist eine PL1-Wahrheit,
genau dann wenn $\neg P$ PL1-unerfüllbar ist.*

PL1-Wahrheit: Beispiel

Ist $a = a$ eine PL1-Wahrheit?

PL1-Wahrheit: Beispiel II

Ist $\text{Large}(a) \vee \text{Medium}(a) \vee \text{Small}(a)$ eine PL1-Wahrheit?

PL1-Wahrheit: Beispiel III

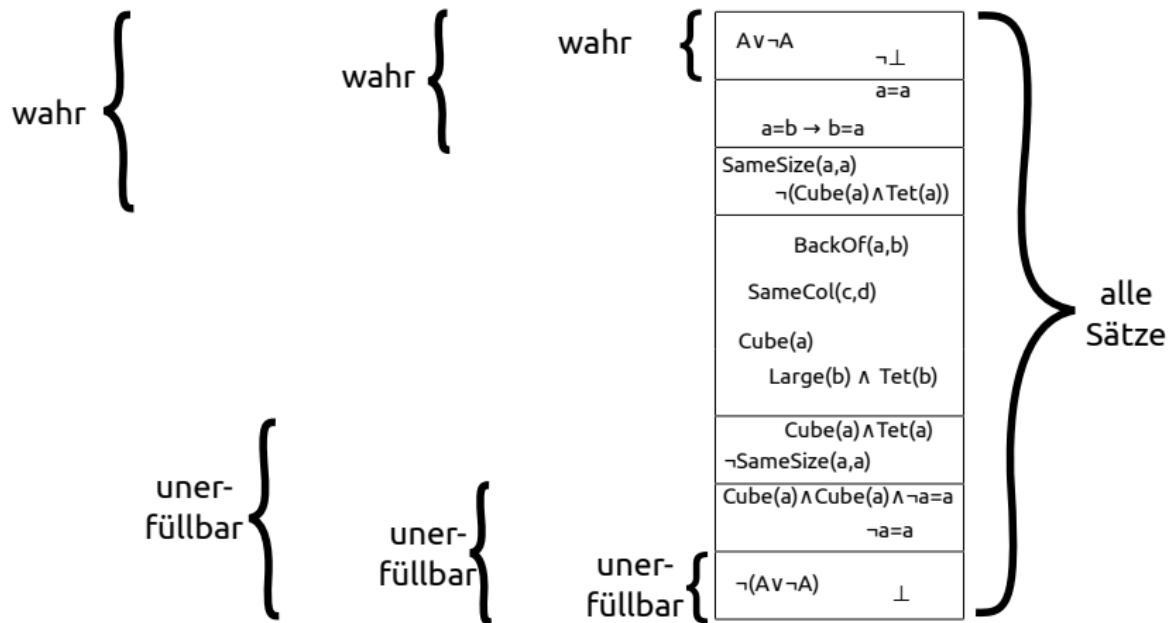
Ist $(a = b \wedge Large(a)) \rightarrow Large(b)$ eine PL1-Wahrheit?

Die verschiedenen Stufen der Wahrheiten und (Un)Erfüllbarkeiten

Bivalence World

PL1

Tautologisch

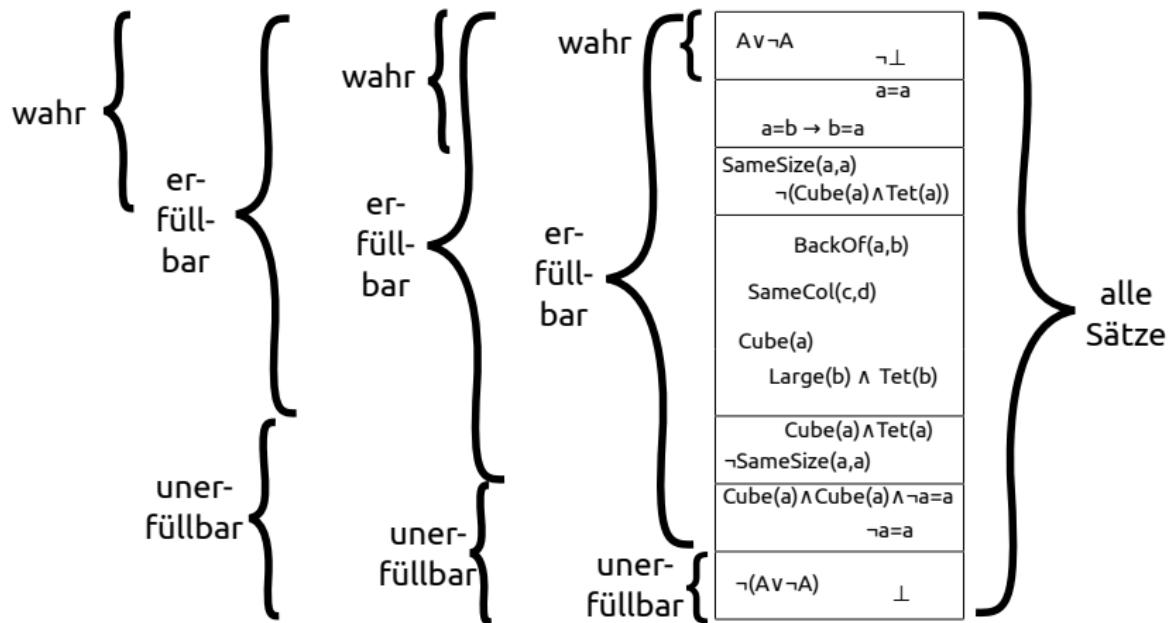


Die verschiedenen Stufen der Wahrheiten und (Un)Erfüllbarkeiten

Bivalence World

PL1

Tautologisch



Lernziele 11

oo

PL1-Strukturen

oooooooooooo

... mit Funktionen

oooooooooooo

PL1-Wahrheit

oooooooooooo

PL1-Folgerung

●oooo

Zusammenfassung

oo

PL1-Folgerung

PL1-Folgerung

Definition

Ein Satz P heißt eine **PL1-Folgerung** aus einer logischen Theorie \mathcal{T} (in Zeichen: $\mathcal{T} \models_{PL1} P$) genau dann, wenn jede PL1-Struktur, die alle Sätze aus \mathcal{T} erfüllt, auch den Satz P erfüllt.

Ein **Gegenbeispiel** zum Vorliegen einer solchen PL1-Folgerung ist eine PL1-Struktur, in der P falsch ist, und alle Sätze aus \mathcal{T} wahr. Um zu **prüfen**, ob P eine PL1-Folgerung aus den Prämissen \mathcal{T} ist, versucht man, ein Gegenbeispiel zu finden. Falls es ein solches gibt, handelt es sich bei dem ursprünglichen Argument um keine PL1-Folgerung. Falls es kein Gegenbeispiel gibt, ist das Argument eine PL1-Folgerung.

Bemerkung: Folgerung und Wahrheit werden beide mit \models notiert:

Folgerung $\mathcal{T} \models_{PL1} P$ (\mathcal{T} : logische Theorie)

versus Wahrheit $\mathfrak{M} \models P$ (\mathfrak{M} : PL1-Struktur).

Ist das ein gültiges PL1-Argument?

Large(a)
¬Medium(a)

Ist das ein gültiges PL1-Argument?

| Tet(a) \wedge Large(a)
| a = b
| Tet(b)

Tautologien und logische Wahrheiten (Fortsetzung)

Aussagenlogik	PL1	Bivalence World	Allgemein (nur informell)
Tautologie	PL1-Wahrheit	BW-Wahrheit	Logische Wahrheit
Tautologische Folgerung	PL1-Folgerung	BW-Folgerung	Logische Folgerung
Tautologische Äquivalenz	PL1-Äquivalenz	BW-Äquivalenz	Logische Äquivalenz
Weltbegriff			
Wahrheitsbelegungen	PL1-Strukturen	BW-Welten	Welten

Zusammenfassung

Veranstaltung 11

Zusammenfassung Veranstaltung 11 und Ausblick

Was haben wir heute gelernt?

- PL1-Strukturen und den Tarski'schen Wahrheitsbegriff
- PL1-Wahrheit, PL1-Erfüllbarkeit, PL1-Folgerung
- den präzisen Unterschied zwischen PL1-Folgerung und anderen Folgerungsbegriffen

Nächstes Mal behandeln wir:

- die PL1-Semantik die Quantoren
- die axiomatische Methode und einige Axiomensysteme

Lernziele Veranstaltung 12

Lernziele für heute (Veranstaltung 12)

- ich lerne die **formale Semantik von Quantoren** kennen.
- ich lerne **Äquivalenzen für Quantoren** kennen
- ich lerne die **axiomatische Methode** kennen
und auf Bivalence World anzuwenden

Buch: 10.1–10.5, 12.5, 18.1, 18.2, 18.4 (letztere aus Band 2)

Formale Semantik der Quantoren

Wahrheit quantifizierter Sätze (A. Tarski)

Fortsetzung der Def. $\mathfrak{M} \models P$ (Satz P ist in der Struktur \mathfrak{M} wahr).

Wir schreiben: Syntax/Objektsprache in **rot** und **lila**,
 Semantik/Metasprache in schwarz und **grün**.

Naiver Ansatz (der leider **nicht** funktioniert!!!):

- 8 $\mathfrak{M} \models \forall x P$ g.d.w. $\mathfrak{M} \models P[x \mapsto u]$ für alle $u \in D^{\mathfrak{M}}$,
- 9 $\mathfrak{M} \models \exists x P$ g.d.w. $\mathfrak{M} \models P[x \mapsto u]$ für ein $u \in D^{\mathfrak{M}}$.

Hierbei bezeichnet $P[x \mapsto u]$ die Formel P , wobei alle freien Vorkommen von x durch u ersetzt werden.

Problem: $P[x \mapsto u]$ mischt

- **Syntax:** Formel P mit
- **Semantik:** Element u aus PL1-Struktur \mathfrak{M}

Das entspricht nicht der Syntax von Sätzen!

Zum Beispiel für $u = \blacksquare$ erhalten wir $P(\blacksquare)$ — das ist keine Formel! (\blacksquare ist kein Element der Syntax.)

Wahrheit quantifizierter Sätze (A. Tarski)

Lösung des Problems: führe neue Konstanten c_i ein!

Definition ($\mathfrak{M} \models P$ (Satz P in der Struktur \mathfrak{M} wahr oder erfüllt))

- ⑧ $\mathfrak{M} \models \forall x P$ g.d.w. $\mathfrak{M}[c_i \mapsto u] \models P[x \mapsto c_i]$ für alle $u \in D^{\mathfrak{M}}$,
- ⑨ $\mathfrak{M} \models \exists x P$ g.d.w. $\mathfrak{M}[c_i \mapsto u] \models P[x \mapsto c_i]$ für ein $u \in D^{\mathfrak{M}}$.

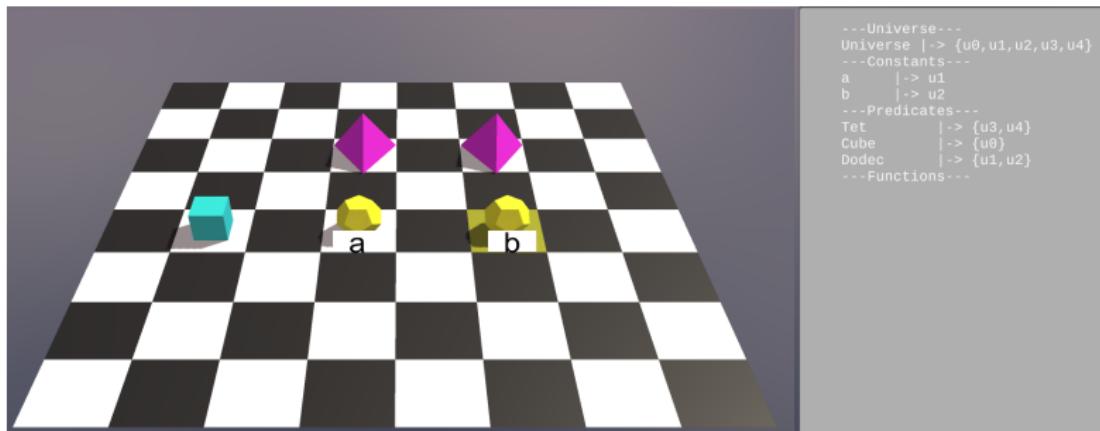
Dabei bezeichnet

- $P[x \mapsto c_i]$ (**Substitution**) die Formel P , wobei alle freien Vorkommen von x durch c_i ersetzt werden,
 - Beispiel: $\text{Larger}(a, y)[y \mapsto c_1]$ ist $\text{Larger}(a, c_1)$
- $\mathfrak{M}[c_i \mapsto u]$ (**Expansion**) das Modell, dass die Konstante c_i mit u interpretiert und sich sonst verhält wie \mathfrak{M} .

c_i muss dabei eine neue, in der Sprache von \mathfrak{M} noch nicht benutzte Konstante sein (vgl. die neuen Konstanten n_1, n_2, \dots , die Bivalence World im Henkin-Hintikka-Spiel einführt).

Beispiel mit einem Quantor

Gilt $\exists x \text{Cube}(x)$ in folgender Bivalence Word-Welt?



Zugehörige PL1-Struktur:

$$D^{\mathfrak{M}} = \{u0, u1, u2, u3, u4\} \quad \quad \mathfrak{M}(a) = u1 \quad \quad \mathfrak{M}(b) = u2$$

$$\mathfrak{M}(\textcolor{red}{Tet}) = \{u3, u5\} \quad \mathfrak{M}(\textcolor{red}{Cube}) = \{u0\} \quad \mathfrak{M}(\textcolor{red}{Dodec}) = \{u1, u2\}$$

$$\mathfrak{M} \models \exists x \text{Cube}(x)$$

Beispiel, Fortsetzung

$\mathfrak{M} \models \exists x \text{Cube}(x)$

g.d.w. für ein $u \in D^{\mathfrak{M}}$: $\mathfrak{M}[c_1 \mapsto u] \models \text{Cube}(x)[x \mapsto c_1]$

g.d.w. für ein $u \in D^{\mathfrak{M}}$: $\mathfrak{M}[c_1 \mapsto u] \models \text{Cube}(c_1)$

g.d.w. für ein $u \in D^{\mathfrak{M}}$ mit $\mathfrak{M}' = \mathfrak{M}[c_1 \mapsto u]$:

$\mathfrak{M}'(c_1) \in \mathfrak{M}'(\text{Cube})$

g.d.w. für ein $u \in D^{\mathfrak{M}}$: $u \in \mathfrak{M}(\text{Cube})$

g.d.w. für ein $u \in \{u0, u1, u2, u3, u4\}$: $u \in \{u0\}$

Da dies gilt für $u \mapsto u0$, ist obiger Satz in \mathfrak{M} wahr.

Beispiel mit anderem Satz

$\mathfrak{M} \models \forall x Dodec(x)$

g.d.w. für alle $u \in D^{\mathfrak{M}}$: $\mathfrak{M}[c_1 \mapsto u] \models Dodec(x)[x \mapsto c_1]$

g.d.w. für alle $u \in D^{\mathfrak{M}}$: $\mathfrak{M}[c_1 \mapsto u] \models Dodec(c_1)$

g.d.w. für alle $u \in D^{\mathfrak{M}}$ mit $\mathfrak{M}' = \mathfrak{M}[c_1 \mapsto u]$:

$\mathfrak{M}'(c_1) \in \mathfrak{M}'(Dodec)$

g.d.w. für alle $u \in D^{\mathfrak{M}}$: $u \in \mathfrak{M}(Dodec)$

g.d.w. für alle $u \in \{u0, u1, u2, u3, u4\}$: $u \in \{u1, u2\}$

Da dies nicht gilt für $u \mapsto u0$ noch für $u \mapsto u3$ noch für $u \mapsto u4$, ist obiger Satz in \mathfrak{M} falsch.

Beispiel II mit einem Quantor

$$D^{\mathfrak{M}} = \{anna, berta, cyrel\}$$

$$\mathfrak{M}(\textcolor{red}{Mag}) = \{(anna, anna), (anna, berta), (cyrel, anna)\}$$

$$\mathfrak{M}(\textcolor{red}{a}) = anna$$

$$\mathfrak{M}(\textcolor{red}{b}) = berta$$

$$\mathfrak{M} \models^? \exists y (Mag(a, y) \wedge \neg Mag(y, y))$$

Beispiel II, Fortsetzung

$$\mathfrak{M} \models \exists y (Mag(a, y) \wedge \neg Mag(y, y))$$

g.d.w. für ein $u \in D^{\mathfrak{M}}$: $\mathfrak{M}[c_1 \mapsto u] \models (\text{Mag}(a, y) \wedge \neg \text{Mag}(y, y)) [y \mapsto c_1]$

g.d.w. für ein $u \in D^{\mathfrak{M}}$: $\mathfrak{M}[c_1 \mapsto u] \models Mag(a, c_1) \wedge \neg Mag(c_1, c_1)$

g.d.w. für ein $u \in D^{\mathfrak{M}}$:

$\mathfrak{M}[c_1 \mapsto u] \models Mag(a, c_1)$ und $\mathfrak{M}[c_1 \mapsto u] \models \neg Mag(c_1, c_1)$

g.d.w. für ein $u \in D^{\mathfrak{M}}$:

$\mathfrak{M}[c_1 \mapsto u] \models Mag(a, c_1)$ und $\mathfrak{M}[c_1 \mapsto u] \not\models Mag(c_1, c_1)$

g.d.w. für ein $u \in D^{\mathfrak{M}}$ mit $\mathfrak{M}' = \mathfrak{M}[\textcolor{red}{c_1} \mapsto u]$:

$(\mathfrak{M}'(a), \mathfrak{M}'(c_1)) \in \mathfrak{M}'(\textcolor{violet}{Mag})$ und $(\mathfrak{M}'(c_1), \mathfrak{M}'(c_1)) \notin \mathfrak{M}'(\textcolor{violet}{Mag})$

g.d.w. für ein $u \in D^{\mathfrak{M}}$: $(\textcolor{red}{anna}, u) \in \mathfrak{M}(\textcolor{red}{Mag})$ und $(\textcolor{blue}{u}, u) \notin \mathfrak{M}(\textcolor{red}{Mag})$

g.d.w. für ein $u \in \{anna, berta, cyrel\}$:

$(anna, u) \in \{(anna, anna), (anna, berta), (cyrel, anna)\}$ und
 $(u, u) \notin \{(anna, anna), (anna, berta), (cyrel, anna)\}$

Da dies gilt für $u \mapsto b_{\sigma(u)}$, ist obiger Satz in \mathfrak{M} wahr.

Beispiele mit zwei Quantoren

$$D^{\mathfrak{M}} = \{anna, berta, cyrel\}$$

$$\mathfrak{M}(\textcolor{red}{Mag}) = \{(anna, anna), (anna, berta), (cyrel, anna)\}$$

$$\mathfrak{M}(a) = anna$$

$$\mathfrak{M}(b) = berta$$

$$\mathfrak{M} \stackrel{?}{\models} \exists x \exists y (Mag(x, y) \wedge \neg Mag(y, y))$$

$$\mathfrak{M} \stackrel{?}{\models} \forall x \exists y (Mag(x, y) \wedge \neg Mag(y, y))$$

Beispiel, Fortsetzung

$$\mathfrak{M} \models \exists x \exists y (Mag(x, y) \wedge \neg Mag(y, y))$$

g.d.w. für ein $u_1 \in D^{\mathfrak{M}}$: $\mathfrak{M}[c_1 \mapsto u_1] \models \exists y (Mag(c_1, y) \wedge \neg Mag(y, y))$

g.d.w. für ein $u_1 \in D^{\mathfrak{m}}$, für ein $u_2 \in D^{\mathfrak{m}}$:

$$\mathfrak{M}[c_1 \mapsto u_1, c_2 \mapsto u_2] \models Mag(c_1, c_2) \wedge \neg Mag(c_2, c_1)$$

g.d.w. für ein $u_1 \in D^{\mathfrak{M}}$, für ein $u_2 \in D^{\mathfrak{M}}$ mit $\mathfrak{M}' = \mathfrak{M}[\textcolor{red}{c_1} \mapsto u_1, \textcolor{red}{c_2} \mapsto u_2]$:

$\mathfrak{M}' \models Mag(c_1, c_2)$ und $\mathfrak{M}' \models \neg Mag(c_2, c_2)$

g.d.w. für ein $u_1 \in D^{\mathfrak{M}}$, für ein $u_2 \in D^{\mathfrak{M}}$ mit $\mathfrak{M}' = \mathfrak{M}[\textcolor{red}{c_1} \mapsto u_1, \textcolor{red}{c_2} \mapsto u_2]$:

$\mathfrak{m}' \models Mag(c_1, c_2)$ und $\mathfrak{m}' \not\models Mag(c_2, c_2)$

g.d.w. für ein $u_1 \in D^{\mathfrak{M}}$, für ein $u_2 \in D^{\mathfrak{M}}$ mit $\mathfrak{M}' = \mathfrak{M}[\textcolor{red}{c_1} \mapsto u_1, \textcolor{red}{c_2} \mapsto u_2]$:

$(\mathfrak{M}'(c_1), \mathfrak{M}'(c_2)) \in \mathfrak{M}(\textcolor{violet}{Mag})$ und $(\mathfrak{M}'(c_2), \mathfrak{M}'(c_2)) \notin \mathfrak{M}(\textcolor{violet}{Mag})$

g.d.w. für ein $u_1 \in D^{\mathfrak{m}}$, für ein $u_2 \in D^{\mathfrak{m}}$:

$(u_1, u_2) \in \mathfrak{M}(\textcolor{red}{Mag})$ und $(u_2, u_2) \notin \mathfrak{M}(\textcolor{red}{Mag})$

g.d.w. für ein $u_1 \in \{anna, berta, cyrel\}$, für ein $u_2 \in \{anna, berta, cyrel\}$:

$(u_1, u_2) \in \{(anna, anna), (anna, berta), (cyrel, anna)\}$ und

$$(u_2, u_2) \notin \{(anna, anna), (anna, berta), (cyrel, anna)\}$$

Da dies gilt für $u_1 \mapsto anna$ und $u_2 \mapsto berta$, ist obiger Satz in \mathfrak{M} wahr.

Erfüllungsinvarianz

Theorem (Erfüllungsinvarianz)

Es seien \mathfrak{M}_1 und \mathfrak{M}_2 Strukturen mit demselben Gegenstandsbereich, zudem ordnen sie den Prädikatssymbolen, Funktionssymbolen und Konstanten in einem Satz P dieselben Interpretationen zu. Dann gilt

$$\mathfrak{M}_1 \models P \text{ genau dann, wenn } \mathfrak{M}_2 \models P.$$

PL1-Erfüllbarkeit und PL1-Wahrheit mit Quantoren

PL1-Erfüllbarkeit mit Quantoren

Wir wiederholen:

Definition

Eine logische Theorie \mathcal{T} heißt **PL1-erfüllbar** genau dann, wenn es eine PL1-Struktur gibt, die alle Sätze aus \mathcal{T} erfüllt.

Beispiel:

$\exists x \text{Cube}(x)$ ist PL1-erfüllbar (siehe Beispiel von Folie 388)

PL1-Erfüllbarkeit: Beispiel

Ist $\neg \text{Cube}(a) \wedge \forall x \text{ Cube}(x)$ PL1-erfüllbar?

PL1-Wahrheit mit Quantoren

Wir wiederholen:

Definition

Ein Satz P heißt eine **PL1-Wahrheit** genau dann, wenn jede PL1-Struktur den Satz P erfüllt.

Beispiel:

$\neg \exists x \text{Cube}(x)$ ist keine PL1-Wahrheit.

- Folie 388 zeigt eine PL1-Struktur, in der $\exists x \text{Cube}(x)$ wahr ist und damit $\neg\exists x \text{Cube}(x)$ falsch.

PL1-Wahrheit: Beispiel

Ist $\forall x \ x = x$ eine PL1-Wahrheit?

PL1-Wahrheit: Beispiel

Ist $\forall x \text{SameSize}(x, x)$ eine PL1-Wahrheit?

Tautologien vs. PL1-Wahrheiten

$$\exists x \text{ Cube}(x) \vee \exists x \neg \text{Cube}(x)$$

ist eine PL1-Wahrheit, jedoch

$$\forall x \text{ Cube}(x) \vee \forall x \neg \text{Cube}(x)$$

nicht. Im Gegensatz dazu ist

$$\forall x \text{ Cube}(x) \vee \neg \forall x \text{ Cube}(x)$$

eine Tautologie.

Wahrheitsfunktionale Form

Ersetze alle quantifizierten Teilformeln, die nicht im Skopus eines anderen Quantors liegen, durch Satzbuchstaben.

Gleiche wohlgeformte Formeln innerhalb eines Satzes werden dabei durch den gleichen Buchstaben ersetzt.

Ein quantifizierter Satz der Sprache erster Stufe heißt genau dann eine **Tautologie**, wenn seine wahrheitsfunktionale Form eine Tautologie ist.

$$\forall x \text{ } \textit{Cube}(x) \vee \neg \forall x \text{ } \textit{Cube}(x)$$

wird zu

$$A \vee \neg A$$

Wahrheitsfunktionale Form – Beispiele

PL1-Satz	w. f. Form
$\forall x \text{Cube}(x) \vee \neg \forall x \text{Cube}(x)$	$A \vee \neg A$
$(\exists y \text{Tet}(y) \wedge \forall z \text{Small}(z)) \rightarrow \forall z \text{Small}(z)$	$(A \wedge B) \rightarrow B$
$\forall x \text{Cube}(x) \vee \exists y \text{Tet}(y)$	$A \vee B$
$\forall x \text{Cube}(x) \rightarrow \text{Cube}(a)$	$A \rightarrow B$
$\forall x (\text{Cube}(x) \vee \neg \text{Cube}(x))$	A
$\forall x (\text{Cube}(x) \rightarrow \text{Small}(x)) \vee \exists x \text{Dodec}(x)$	$A \vee B$

Beispiele für die \rightarrow -Elimination
$$\exists x (\text{Cube}(x) \rightarrow \text{Small}(x))$$
$$\exists x \text{ Cube}(x)$$
$$\exists x \text{ Small}(x)$$
$$A$$
$$B$$
$$C$$

Nein!

$$\exists x \text{Cube}(x) \rightarrow \exists x \text{ Small}(x)$$
$$\exists x \text{ Cube}(x)$$
$$\exists x \text{ Small}(x)$$
$$A \rightarrow B$$
$$A$$
$$B$$

Ja!

Tautologien und PL1-Wahrheiten

Theorem

Jede Tautologie ist eine PL1-Wahrheit, aber nicht umgekehrt.

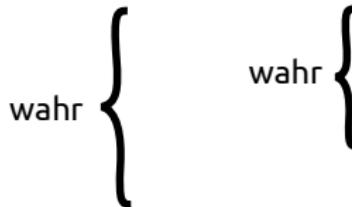
Beispiel:

$$\exists x \text{ Cube}(x) \vee \exists x \neg \text{Cube}(x)$$

ist eine PL1-Wahrheit, aber keine Tautologie.

Die verschiedenen Stufen der Wahrheiten und (Un)Erfüllbarkeiten

Bivalence World



PL1



Tautologisch

wahr {

$P \vee \neg P$	\perp
$\neg \perp$	$a = a$
$\exists x \text{Cube}(x) \vee \exists x \neg \text{Cube}(x)$	$\neg (\text{Cube}(a) \wedge \text{Tet}(a))$
$\neg (\text{Cube}(a) \wedge \text{Tet}(a))$	$\exists x \text{ BackOf}(x,a)$
$\exists x \text{ BackOf}(x,a)$	$\exists y \forall x \text{ SameCol}(x,y)$
$\exists y \forall x \text{ SameCol}(x,y)$	$\text{Cube}(a)$
$\text{Cube}(a)$	$\forall x \text{ Tet}(x)$
$\forall x \text{ Tet}(x)$	$\text{Cube}(a) \wedge \text{Tet}(a)$
$\text{Cube}(a) \wedge \text{Tet}(a)$	$\neg \text{SameSize}(a,a)$
$\neg \text{SameSize}(a,a)$	$\forall x \text{Cube}(x) \wedge \forall x \neg \text{Cube}(x)$
$\forall x \text{Cube}(x) \wedge \forall x \neg \text{Cube}(x)$	$\neg a = a$
$\neg a = a$	$\neg (P \vee \neg P)$
$\neg (P \vee \neg P)$	\perp

alle Sätze

unerfüllbar {

unerfüllbar {

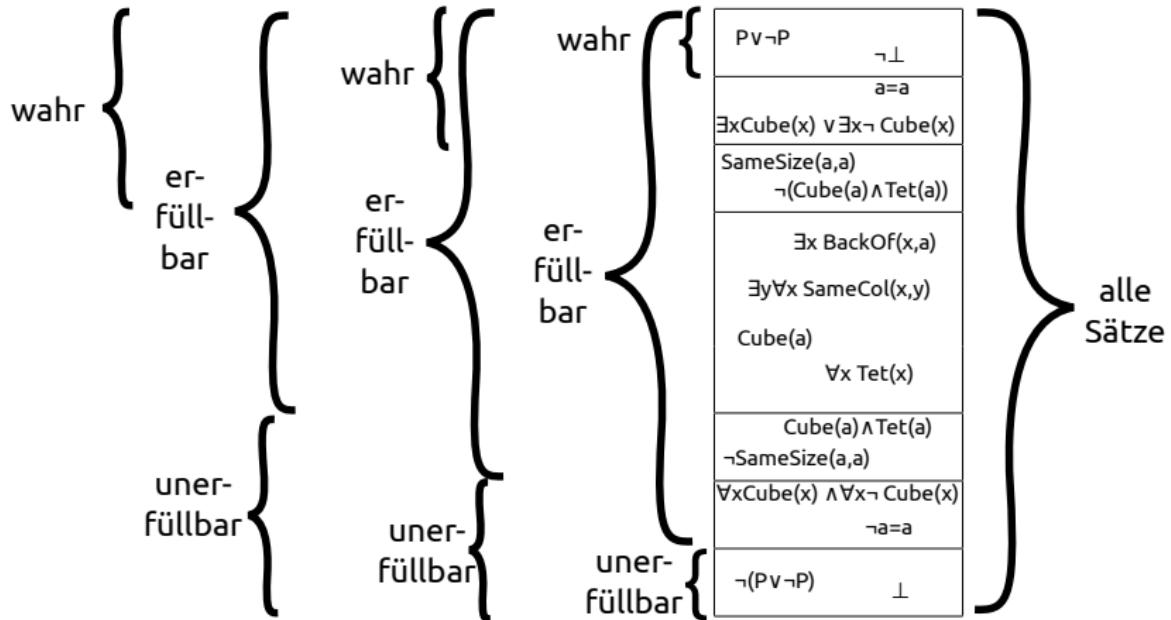
unerfüllbar {

Die verschiedenen Stufen der Wahrheiten und (Un)Erfüllbarkeiten

Bivalence World

PL1

Tautologisch



PL1-Folgerung mit Quantoren

PL1-Folgerung

Wir wiederholen:

Definition

Ein Satz P heißt eine **PL1-Folgerung** aus einer logischen Theorie \mathcal{T} (in Zeichen: $\mathcal{T} \models_{PL1} P$) genau dann, wenn jede PL1-Struktur, die alle Sätze aus \mathcal{T} erfüllt, auch den Satz P erfüllt.

Ist das ein gültiges PL1-Argument?

$$\begin{array}{c} \forall x \text{Cube}(x) \\ \hline \neg \exists x \text{Tet}(x) \end{array}$$

Ist das ein gültiges PL1-Argument?

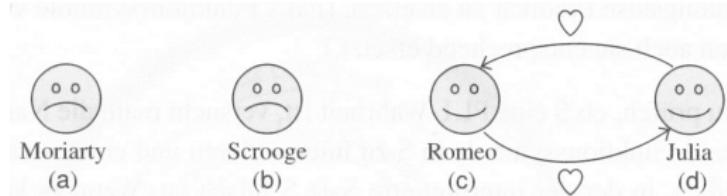
$\neg \exists x \text{ Larger}(x, a)$

$\neg \exists x \text{ Larger}(b, x)$

$\text{Larger}(c, d)$

$\text{Larger}(a, b)$

Ein Gegenbeispiel



Ein PL1-Gegenbeispiel – Interpretiere $Larger(x, y)$ als $Mag(x, y)$

- $D^{\mathfrak{M}} = \{moriarty, scrooge, romeo, julia\}$,
- $\mathfrak{M}(a) = moriarty$,
 $\mathfrak{M}(b) = scrooge$,
- $\mathfrak{M}(c) = romeo$,
- $\mathfrak{M}(d) = julia$,
- $\mathfrak{M}(Larger) = \{(romeo, julia), (julia, romeo)\}$.

Ist das ein gültiges PL1-Argument?

| $\forall x(Tet(x) \rightarrow Large(x))$
| $\neg Large(b)$
| $\neg Tet(b)$

Tautologische und PL1-Folgerungen

Theorem

Jede tautologische Folgerung ist auch eine PL1-Folgerung, aber nicht umgekehrt.

Beispiel:

$$\frac{}{\forall x \text{ Cube}(x)} \quad \frac{}{\exists x \text{ Cube}(x)}$$

ist eine PL1-Folgerung, aber keine tautologische Folgerung.

Weitere logische Folgerungen für Quantoren

$$\begin{array}{c}
 1 \quad \left| \begin{array}{l} \forall x(\text{Cube}(x) \rightarrow \text{Small}(x)) \\ \forall x \text{ Cube}(x) \\ \hline \forall x \text{ Small}(x) \end{array} \right.
 \end{array}$$

$$\begin{array}{c}
 3 \quad \left| \begin{array}{l} \exists x(\text{Cube}(x) \rightarrow \text{Small}(x)) \\ \exists x \text{ Cube}(x) \\ \hline \exists x \text{ Small}(x) \end{array} \right.
 \end{array}$$

$$\begin{array}{c}
 2 \quad \left| \begin{array}{l} \forall x \text{ Cube}(x) \\ \forall x \text{ Small}(x) \\ \hline \forall x(\text{Cube}(x) \wedge \text{Small}(x)) \end{array} \right.
 \end{array}$$

$$\begin{array}{c}
 4 \quad \left| \begin{array}{l} \exists x \text{ Cube}(x) \\ \exists x \text{ Small}(x) \\ \hline \exists x(\text{Cube}(x) \wedge \text{Small}(x)) \end{array} \right.
 \end{array}$$

Welche Argumente sind gültig (PL1-Folgerungen)?

PL1-Äquivalenz

PL1-Äquivalenzen

Definition

Zwei Sätze P und Q heißen **PL1-äquivalent**, wenn sie in denselben PL1-Strukturen wahr sind. In Zeichen:

$$P \Leftrightarrow_{PL1} Q$$

Substitutionsprinzip

Wenn $P \Leftrightarrow_{PL1} Q$, dann $S(P) \Leftrightarrow_{PL1} S(Q)$.

Hierbei ist $S(_)$ ein Satz mit einer „Lücke“.

Beispiel:

$A \wedge B \Leftrightarrow_{PL1} B \wedge A$, daher auch $C \rightarrow (A \wedge B) \Leftrightarrow_{PL1} C \rightarrow (B \wedge A)$.

Hier ist $S(_)$ der Satz $C \rightarrow (_)$.

DeMorgansche Gesetze für Quantoren

$$\neg \forall x P(x) \Leftrightarrow_{PL1} \exists x \neg P(x)$$

$$\forall x P(x) \Leftrightarrow_{PL1} \neg \exists x \neg P(x)$$

$$\neg \exists x P(x) \Leftrightarrow_{PL1} \forall x \neg P(x)$$

$$\exists x P(x) \Leftrightarrow_{PL1} \neg \forall x \neg P(x)$$

Weitere Äquivalenzen für Quantoren (1)

$$\forall x (P(x) \wedge Q(x)) \Leftrightarrow_{PL1} \forall x P(x) \wedge \forall x Q(x)$$

$$\forall x (P(x) \vee Q(x)) \not\Leftrightarrow_{PL1} \forall x P(x) \vee \forall x Q(x)$$

$$\exists x (P(x) \vee Q(x)) \Leftrightarrow_{PL1} \exists x P(x) \vee \exists x Q(x)$$

$$\exists x (P(x) \wedge Q(x)) \not\Leftrightarrow_{PL1} \exists x P(x) \wedge \exists x Q(x)$$

Weitere Äquivalenzen für Quantoren (2)

Falls x in P nicht frei ist, gelten

$$\forall x P \quad \Leftrightarrow_{PL1} \quad P$$

$$\exists x P \quad \Leftrightarrow_{PL1} \quad P$$

$$\forall x (P \vee Q(x)) \quad \Leftrightarrow_{PL1} \quad P \vee \forall x Q(x)$$

$$\exists x (P \wedge Q(x)) \quad \Leftrightarrow_{PL1} \quad P \wedge \exists x Q(x)$$

Weitere Äquivalenzen für Quantoren (3)

Falls y in $P(x)$ nicht vorkommt, gelten

$$\forall x P(x) \Leftrightarrow_{PL1} \forall y P(y)$$

$$\exists x P(x) \Leftrightarrow_{PL1} \exists y P(y)$$

Die axiomatische Methode

BW Folgerung \neq PL1-Folgerung

Wir haben Argumente gefunden, die in Bivalence World gültig sind, aber nicht PL1-gültig.

$$\begin{array}{l} \vdash \forall x (\text{Cube}(x) \leftrightarrow \text{SameShape}(x, c)) \\ \vdash \text{Cube}(c) \end{array}$$

Die axiomatische Methode

Die [axiomatische Methode](#) überbrückt die Lücke zwischen der Bivalence World-Folgerung und der PL1-Folgerung, indem man in systematischer Weise Tatsachen über die Prädikatsymbole, die in unseren Argumenten vorkommen, zum Ausdruck bringt.

Die Sätze, die diese Tatsachen ausdrücken, werden als **Axiome** eingeführt. Sie bilden zusammen eine **logische Theorie** oder **Axiomensystem** und beschränken die mögliche Interpretation der Prädikatsymbole.

Axiome können als zusätzliche Prämissen bei Folgerungen und Beweisen genutzt werden.

Das Argument noch einmal betrachtet

Mit einer zusätzlichen Prämissen wird das Argument gültig:

- $\forall x (\text{Cube}(x) \leftrightarrow \text{SameShape}(x, c))$
 - $\forall x \text{SameShape}(x, x)$
 - | $\text{Cube}(c)$

Die grundlegenden Formaxiome

- ① $\neg \exists x (Cube(x) \wedge Tet(x))$
 - ② $\neg \exists x (Tet(x) \wedge Dodec(x))$
 - ③ $\neg \exists x (Dodec(x) \wedge Cube(x))$
 - ④ $\forall x (Tet(x) \vee Dodec(x) \vee Cube(x))$

Ein Argument, welches die Formaxiome nutzt

- $\neg \exists x (\text{Dodec}(x) \wedge \text{Cube}(x))$
- $\forall x (\text{Tet}(x) \vee \text{Dodec}(x) \vee \text{Cube}(x))$
- $\neg \exists x \text{ Tet}(x)$
- $\forall x (\text{Cube}(x) \leftrightarrow \neg \text{Dodec}(x))$

Einführungs- und Beseitigungsaxiome von SameShape

- ① $\forall x \forall y ((Cube(x) \wedge Cube(y)) \rightarrow SameShape(x, y))$
- ② $\forall x \forall y ((Dodec(x) \wedge Dodec(y)) \rightarrow SameShape(x, y))$
- ③ $\forall x \forall y ((Tet(x) \wedge Tet(y)) \rightarrow SameShape(x, y))$

- ④ $\forall x \forall y ((SameShape(x, y) \wedge Cube(x)) \rightarrow Cube(y))$
- ⑤ $\forall x \forall y ((SameShape(x, y) \wedge Dodec(x)) \rightarrow Dodec(y))$
- ⑥ $\forall x \forall y ((SameShape(x, y) \wedge Tet(x)) \rightarrow Tet(y))$

Berühmte Axiomensysteme

- Euklids Axiomatisierung der Geometrie
- Peanos Axiomatisierung der natürlichen Zahlen
- Zermelo-Fraenkel-Axiomatisierung der Mengenlehre
- Axiomatisierungen in der Algebra: Monoid, Gruppe, Ring, Körper, Vektorraum ...
- Hoares Axiomatisierung einer imperativen Sprache mit den Konstrukten Zuweisung von Werten zu Variablen, while-loop, if-then-else,

Semantik von logischen Theorien

Definition

Eine logische Theorie \mathcal{T} definiert eine Klasse von PL1-Strukturen: nämlich diejenigen PL1-Strukturen, die alle Sätze der Theorie erfüllen. Diese Klasse wird auch mit $\text{Mod}(\mathcal{T})$ bezeichnet.

Jedes Axiomensystem ist eine logische Theorie. Insofern gilt diese Definition auch für Axiomensysteme.

$\text{Mod}(\mathcal{T})$ für bekannte Axiomensysteme \mathcal{T} :

- In der Algebra sind dies z.B. Monoide, Gruppen, Ringe etc.
- Für das Axiomensystem partieller Ordnungen sind es alle partiellen Ordnungen.
- Für das Axiomensystem der natürlichen Zahlen sind es die natürlichen Zahlen sowie weitere Zahlenbereiche ("Nicht-Standard-Zahlen").

Alle Formaxiome

- ① $\neg \exists x (Cube(x) \wedge Tet(x))$
- ② $\neg \exists x (Tet(x) \wedge Dodec(x))$
- ③ $\neg \exists x (Dodec(x) \wedge Cube(x))$

- ④ $\forall x (Tet(x) \vee Dodec(x) \vee Cube(x))$

- ⑤ $\forall x \forall y ((Cube(x) \wedge Cube(y)) \rightarrow SameShape(x, y))$
- ⑥ $\forall x \forall y ((Dodec(x) \wedge Dodec(y)) \rightarrow SameShape(x, y))$
- ⑦ $\forall x \forall y ((Tet(x) \wedge Tet(y)) \rightarrow SameShape(x, y))$

- ⑧ $\forall x \forall y ((SameShape(x, y) \wedge Cube(x)) \rightarrow Cube(y))$
- ⑨ $\forall x \forall y ((SameShape(x, y) \wedge Dodec(x)) \rightarrow Dodec(y))$
- ⑩ $\forall x \forall y ((SameShape(x, y) \wedge Tet(x)) \rightarrow Tet(y))$

Vollständigkeit der Formaxiome

Definition

Zwei Strukturen heißen **isomorph**, wenn es eine Bijektion zwischen ihren Gegenstandsbereichen gibt, die verträglich mit ihren Extensionen von Prädikatsymbolen und Interpretationen der Individuenkonstanten sind.

Wir betrachten die Sprache *Cube*, *Tet*, *Dodec*, *SameShape*.

Lemma

Für jede Struktur, die die Formaxiome erfüllt, gibt es eine isomorphe Bivalence World-Struktur.

Theorem

Es sei S ein Satz.

Wenn S eine Bivalence World-Folgerung von \mathcal{T} ist, dann ist S auch eine PL1-Folgerung von \mathcal{T} , erweitert um die Formaxiome.

Zusammenfassung Veranstaltung 12

Zusammenfassung Veranstaltung 12 und Ausblick

Was haben wir heute gelernt?

- Semantik von Quantoren
 - Äquivalenzen für Quantoren
 - den Unterschied zwischen tautologischer und PL1-Folgerung
 - die axiomatische Methode

Nächstes Mal behandeln wir:

- Fitch-Beweise für PL1

Lernziele Veranstaltung 13

Lernziele für heute (Veranstaltung 13)

- ich lerne Beweismethoden für Identität und Quantoren kennen
 - ich lerne Antworten auf die Frage kennen: sind die Fitch-Regeln für Quantoren korrekt?

Buch: 12.1–12.4, 13.1–13.3

Beweismethoden

Wiederholung: Beweise

- Ein Beweis zeigt die Gültigkeit eines Arguments
 - Ein Beweis besteht aus einer Folge von **Beweisschritten**.
 - Jeder Beweisschritt muss gültig sein und muss in
 - **informellen** Beweisen bedeutsam aber leicht zu verstehen sein,
 - **formalen** Beweisen durch **Beweisregeln** gezeigt werden.
 - Folgende gültige Schlussprinzipien dürfen in informellen (aber nicht formalen) Beweisen zumeist stillschweigend genutzt werden:
 - Schließe von $P \wedge Q$ auf P .
 - Schließe von P und Q auf $P \wedge Q$.

Formale Beweise in Fitch

- Wir haben eine wohldefinierte Menge **formaler Beweisregeln**.
- Formale Beweise in Fitch können **mechanisch geprüft** werden.
- Für jeden Junktor gibt es
 - eine **Einführungsregel**, z. B. “von P schließe auf $P \vee Q$ ”,
 - eine **Beseitigungsregel**, z. B. “von $P \wedge Q$ schließe auf P ”.

Beweismethoden und formale Beweise für die Identität

Vier wichtige Prinzipien für die Identitätsrelation

- ① = **Elim**: Gilt $b = c$, dann trifft alles, was auf b zutrifft, auch auf c zu ([Ununterscheidbarkeit von Identischem](#)).
 - ② = **Intro**: Sätze der Form $b = b$ sind stets wahr (in einer PL1-Sprache). ([Reflexivität der Identität](#)).
 - ③ **Symmetrie der Identität**: Wenn $b = c$, dann $c = b$.
 - ④ **Transitivität der Identität**: Wenn $a = b$ und $b = c$, dann $a = c$.

Die beiden letzten Prinzipien folgen dabei aus den ersten beiden.

Transitivität ...



**Logic: another thing that
penguins aren't very good at.**

Informeller Beweis der Symmetrie der Identität

- Es sei $a = b$.
 - Es gilt $a = a$, gemäß der Reflexivität der Identität.
 - Nun ersetzen wir das erste Vorkommen des Namens a in $a = a$ durch den Namen b , wobei wir das Prinzip der Ununterscheidbarkeit von Identischem verwenden.
 - Damit ergibt sich wie gewünscht $b = a$.

Formaler Beweis der Symmetrie der Identität

- 1. $a = b$
- 2. $a = a$ $\Rightarrow \text{Intro}$
- 3. $b = a$ $\Rightarrow \text{Elim: 2, 1}$

Fitch-Regel: Identity Introduction (Identitäts-Einführung)

Beweisregel (= Intro)

Identity Introduction (= Intro):

▷ | $n = n$

Fitch-Regel: Identity Elimination (Identitäts-Beseitigung)

Beweisregel (= Elim)

Identity Elimination (= Elim):

$$\begin{array}{c} P(n) \\ \vdots \\ n = m \\ \vdots \\ \triangleright P(m) \end{array}$$

Präzisierung mittels Substitutionen

Wir hatten die **Substitution** $P[x \mapsto n]$ definiert als die wohlgeformte Formel P , wobei alle freien Vorkommen von x durch n ersetzt werden.

- Beispiel: $\text{Larger}(a, y)[y \mapsto n]$ ist $\text{Larger}(a, n)$

Damit können wir die Beweisregel = **Elim** präziser fassen:

Beweisregel (= Elim)

$$P[x \mapsto n]$$

- 3 -

$$n = m$$

- 3 -

▷ P[x ↦ m]

Beispiel-Beweis mit Identität in Fitch

Cube(a)

$$b = c \vee a \neq b$$

$a = b \rightarrow \text{Cube}(c)$

Beweismethoden und formale Beweise für Quantoren

Beweismethoden für Quantoren

Allbeseitigung

Ausgehend von $\forall x S(x)$ kann man auf $S(c)$ schließen, sofern c einen Gegenstand des Gegenstandsbereiches bezeichnet.

Existenzeinführung

Ausgehend von $S(c)$ kann man auf $\exists x S(x)$ schließen, sofern c einen Gegenstand des Gegenstandsbereiches bezeichnet.

Allbeseitigung

Beweisregel (\forall Elim)

Allbeseitigung (\forall Elim)

$$\triangleright \left| \begin{array}{l} \forall x S(x) \\ \vdots \\ S(c) \end{array} \right.$$

Existenzeinführung

Beweisregel (\exists Intro)

Existenzeinführung
(\exists Intro)

$$\triangleright \left| \begin{array}{l} S(c) \\ \vdots \\ \exists x S(x) \end{array} \right.$$

Beispiel: **∀-Elim** and **∃-Intro**

$\forall x [Cube(x) \rightarrow Large(x)]$
 $\forall x [Large(x) \rightarrow LeftOf(x, b)]$
 $Cube(d)$

$\exists x [Large(x) \wedge LeftOf(x, b)]$

Existenzielle Instantiierung (Existenzbeseitigung)

Existenzbeseitigung

Ausgehend von $\exists x S(x)$ können wir in einem Unterbeweis $S(c)$ annehmen, unter der Bedingung, dass c ein neuer Name ist, der noch nicht verwendet wurde.

Beispiel:

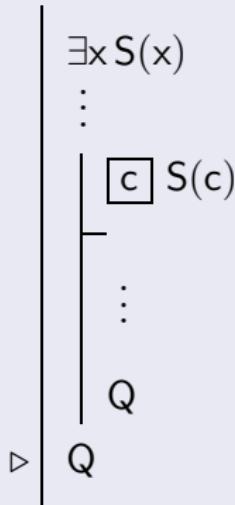
Als Scotland Yard herausfand, dass ein Serienmörder sein Unwesen trieb, nannten sie ihn „**Jack the Ripper**“, und man verwendete diesen Namen, um über ihn Überlegungen anzustellen. Niemand glaubte, dies bedeute, dass Scotland Yard wusste, wer der Mörder war.

Beachten Sie, dass die Verwendung dieses Namens durch Scotland Yard eine schreiende Ungerechtigkeit gewesen wäre, falls der Stadtschneider schon Jack the Ripper genannt worden wäre.

Existenzbeseitigung

Beweisregel (\exists Elim)

Existenzbeseitigung (\exists Elim):



Dabei ist c ein Name, der nicht außerhalb dieses Unterbeweises benutzt wird.

Beispiel: \exists -Elim
$$\begin{array}{l} \forall x [Cube(x) \rightarrow Large(x)] \\ \forall x [Large(x) \rightarrow LeftOf(x, b)] \\ \exists x Cube(x) \end{array}$$
$$\exists x [Large(x) \wedge LeftOf(x, b)]$$

Universelle Generalisierung (Alleinführung)

Alleinführung

Wenn wir einen neuen Namen c einführen, der vorher noch nicht benutzt wurde und dann den Satz $S(c)$ beweisen, können wir auf $\forall x S(x)$ schließen.

Beispiel:

Theorem

Jede positive gerade Zahl ist die Summe von zwei ungeraden Zahlen.

Beweis

Es sei n eine gerade Zahl mit $n > 0$, d. h. $n = 2m$ mit $m > 0$.

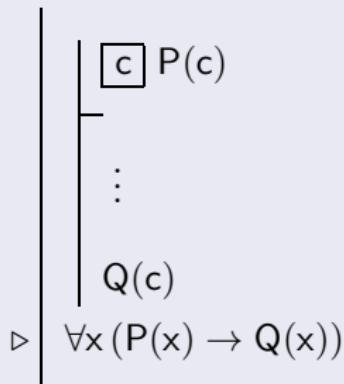
Falls m ungerade ist, dann gilt $n = m + m$, also die Behauptung.

Falls m gerade ist, setzen wir $n = (m - 1) + (m + 1)$.

Allgemeiner konditionaler Beweis

Beweisregel (\forall Intro)

Allgemeiner konditionaler Beweis (\forall Intro):



Dabei ist c ein Name, der nicht außerhalb dieses Unterbeweises benutzt wird.

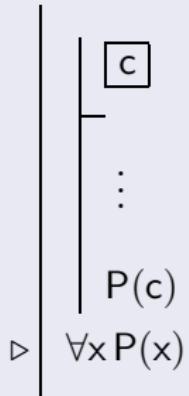
Beispiel: Allgemeiner konditionaler Beweis

$$\begin{array}{|l} \forall x (\text{Cube}(x) \rightarrow \text{Large}(x)) \\ \forall x (\text{Large}(x) \rightarrow \text{LeftOf}(x, b)) \\ \hline \end{array}$$
$$\forall x (\text{Cube}(x) \rightarrow \text{LeftOf}(x, b))$$

Alleinführung

Beweisregel (\forall Intro)

Alleinführung (\forall Intro):



Dabei ist c ein Name, der nicht außerhalb dieses Unterbeweises benutzt wird.

Beispiel: de Morgansches Gesetz

$$\neg \exists x \neg P(x)$$

$$\forall x P(x)$$

Beispiel mit gemischten Quantoren. Wo ist der Fehler?

1. $\forall x \exists y \text{SameCol}(x, y)$

2. \boxed{c}

3. $\exists y \text{SameCol}(c, y)$

\forall Elim: 1

4. $\boxed{d} \text{SameCol}(c, d)$

5. $\text{SameCol}(c, d)$

Reit: 4

6. $\text{SameCol}(c, d)$

\exists Elim: 3, 4–5

7. $\forall x \text{SameCol}(x, d)$

\forall Intro: 2–6

8. $\exists y \forall x \text{SameCol}(x, y)$

\exists Intro: 7

Beispiel mit gemischten Quantoren: korrigiert

$\exists y \forall x \text{SameCol}(x, y)$

$\forall x \exists y \text{SameCol}(x, y)$

Weitere Beweise mit gemischten Quantoren

Welches sind logische Folgerungen?

- ①
 - $\exists y [Cube(y) \wedge \forall x (Dodec(x) \rightarrow SameCol(x, y))]$
 - $\forall x [Dodec(x) \rightarrow \exists y (Cube(y) \wedge SameCol(x, y))]$

- ②
 - $\forall x [Dodec(x) \rightarrow \exists y (Cube(y) \wedge SameCol(x, y))]$
 - $\exists y [Cube(y) \wedge \forall x (Dodec(x) \rightarrow SameCol(x, y))]$

Beweis mit gemischten Quantoren

$\exists y [Cube(y) \wedge \forall x (Dodec(x) \rightarrow SameCol(x, y))]$

$\forall x [Dodec(x) \rightarrow \exists y (Cube(y) \wedge SameCol(x, y))]$

Korrektheit der PL1

Korrektheit von \mathcal{F} für PL1

Theorem

Wenn $\mathcal{T} \vdash S$ gilt, dann ist S eine PL1-Folgerung aus der logischen Theorie \mathcal{T} .

Beweis.

Durch starke Induktion über die Länge des Beweises

Wir werden zeigen, dass jeder Satz, der in einem Schritt im Rahmen eines Beweises auftritt, eine PL1-Folgerung aus denjenigen Annahmen ist, die zu diesem Schritt in Kraft sind.

Eine Annahme ist in einem Schritt **in Kraft**, falls sie eine Annahme des aktuellen Unterbeweises ist oder eine vorherige Annahme eines Beweises einer höheren Ebene.

Für Regeln, für die Unterbeweise mit neuen Konstanten eingeführt werden müssen, brauchen wir die Erfüllungsinvarianz (Veranstaltung 12). □

Conditional Elimination (→ Elim)

k	P → Q
	:
l	P
	:
m+1 ▷	Q

Existenzbeseitigung (\exists Elim):

j	$\exists x S(x)$
	:
k	$\boxed{c} S(c)$
	:
l	Q
m+1 ▷	Q

Dabei ist c ein Name, der nicht außerhalb dieses Unterbeweises benutzt wird.

Zusammenfassung Veranstaltung 13

Zusammenfassung Veranstaltung 13 und Ausblick

Was haben wir heute gelernt?

- Beweismethoden für Identität und Quantoren
 - den Korrektheitsbeweis für die PL1-Regeln des Fitch-Kalküls

Nächstes Mal behandeln wir:

- eine **domänenspezifische** logische Sprache, und zwar für die Domäne der **Ontologien**
 - Ontologien = logische formalisierte Begriffssysteme

Lernziele Veranstaltung 14

Lernziele für heute (Veranstaltung 14)

- ich lerne eine **Ontologiesprache** kennen
 - gleiche Semantik wie PL1, aber andere Syntax
 - domänenspezifische Sprache für die Formulierung von Ontologien

Literatur:

Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, Peter F. Patel-Schneider: *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, second edition, 2007.

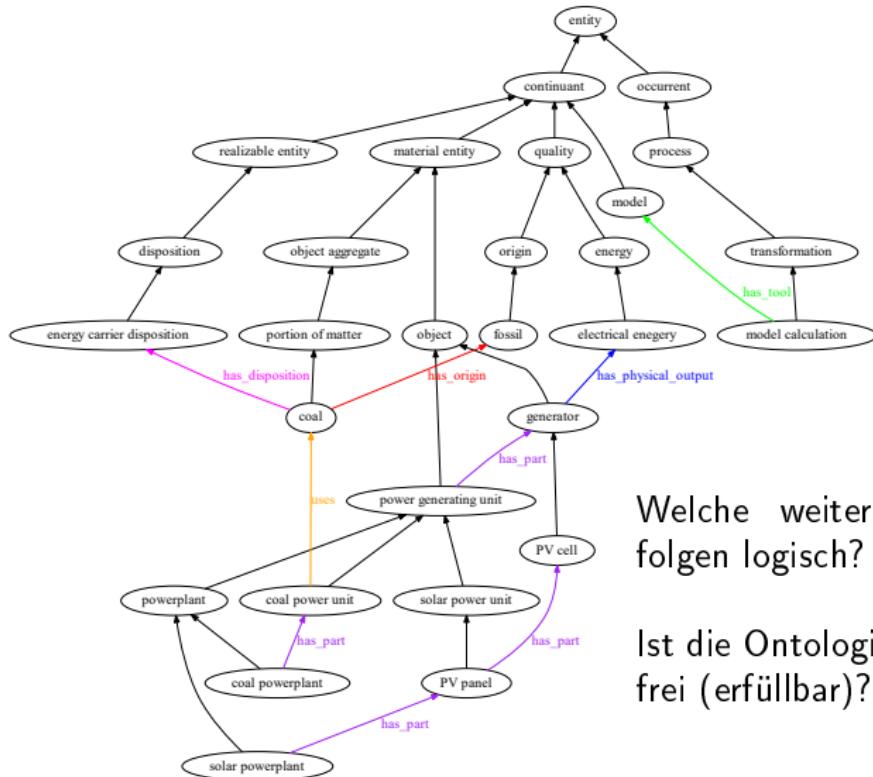
Ontologiesprachen

Ontologien

Ontologien ...

- sind **Begriffsbildungen** für eine Wissensdomäne, die einen **Konsens von Expert:innen** darstellen
- beinhalten eine **Taxonomie** (Hierarchie) von Begriffen (**Klassen**)
- beinhalten **Beziehungen** zwischen Begriffen (Rollen, Relationen)
- umfassen eine **logische Axiomatisierung**
- sind sowohl von **Menschen** als auch von **Maschinen** lesbar
- erlauben automatisches logisches Schließen
 - **Erfüllbarkeit**
 - **logische Folgerungen** (z.B. Erweiterung der Begriffshierarchie)

Wissenrepräsentation in der Open Energy Ontology



Welche weitere Beziehungen folgen logisch?

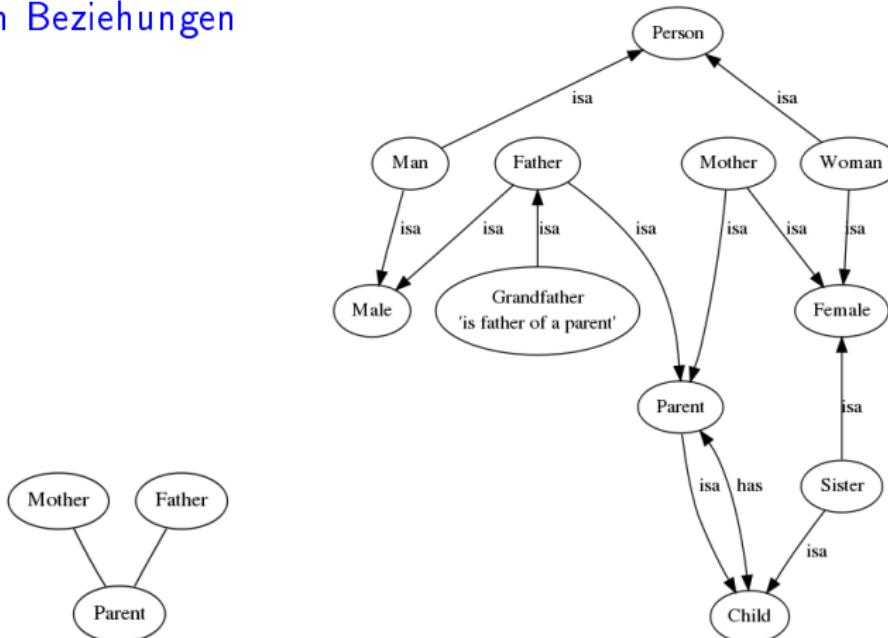
Ist die Ontologie widerspruchsfrei (erfüllbar)?

Logiken für Ontologien

- Logik erster Stufe (PL1)
 - logische Folgerung, Erfüllbarkeit etc. sind **unentscheidbar**
 - PL1 wird für **Grundlagenontologien** (upper ontologies) genutzt
 - standardisierte PL1-Varianten: Common Logic u.a.
- Beschreibungslogiken
 - **effizient entscheidbare** Fragmente von PL1
 - werden für **Domänenontologien** genutzt
 - vom W3C standardisiert als Web ontology language (OWL2 DL)

Semantische Netze

- wurden in 1980ern genutzt für Wissensrepräsentation und logisches Schließen auf diesen Repräsentationen
- z.B. KL-ONE: Schlussfolgern über Begriffe, Unterklassen und deren Beziehungen



Beschreibungslogiken

- Nachteile semantischer Netze:
 - oft ist die Bedeutung der Kanten in dem Graphen nicht präzise definiert
 - manchmal ist die zugrunde liegende Logik **unentscheidbar**
- Beschreibungslogiken
 - haben vollständig **formale Syntax und Semantik** (wie PL1 auch)
 - entscheidbares Fragment von PL1
 - effiziente **Beweistools** verfügbar (Konclude, Pellet, Fact++, Racer)

Wir werden im folgenden die Beschreibungslogik **\mathcal{ALC}** einführen.
Eine Erweiterung von \mathcal{ALC} liegt der Ontologiesprache OWL2 DL zugrunde.

\mathcal{ALC} steht für “attributive language with complement”.

ALC: Konzepte und Individuen

- Konzepte (in OWL: Klassen) (Mother, Father, etc.)
- Subsumption $C \sqsubseteq D$ (lies: "C wird subsumiert durch D")
meint dass jedes C auch ein D ist
 - $Woman \sqsubseteq Person$
 - $Father \sqsubseteq Male$
 - ...
- Individuen peter, mary etc.
 - sind in Konzepten enthalten oder nicht
 - z.B. mary : Person
- Grob gesprochen können Konzepte semantisch auch als Mengen von Individuen aufgefasst werden

ALC: Rollen (zweistellige Relationen)

- Um Konzepte miteinander in Beziehung zu setzen, brauchen wir **Rollen** (in OWL: **properties**):
 - hasChild*
 - partOf*
 - locatedIn*
 - hasColour*
- Zwei Individuen können durch eine Rolle verbunden sein, z.B.
 $\text{hasChild}(\text{mary}, \text{peter})$

Definition

Eine \mathcal{ALC} -Signatur $\Sigma = (\mathbf{C}, \mathbf{R}, \mathbf{I})$ besteht aus

- einer Menge **C** von atomaren Konzepten
- einer Menge **R** von Rollen
- einer Menge **I** von Individuen

ALC: komplexe Konzepte

Definition

Für eine Signatur $\Sigma = (\mathbf{C}, \mathbf{R}, \mathbf{I})$ wird die Menge $\mathbf{Con}(\Sigma)$ der **ALC-Konzepte** über Σ durch die folgende Grammatik definiert:

Manchester-Syntax

$C ::= A$ für $A \in \mathbf{C}$	A
T	Thing
⊥	Nothing
$\neg C$	not C
$C \sqcap C$	C and C
$C \sqcup C$	C or C
$\exists R.C$ for $R \in \mathbf{R}$	R some C
$\forall R.C$ for $R \in \mathbf{R}$	R only C

Als Syntax verwenden wir Manchester-Syntax, weil diese gut für Menschen lesbar ist.

ALC: Beispielkonzept

Ein graues Säugetier mit einem Rüssel.

Ein Säugetier, das einen Rüssel als Körperteil hat und dessen einzige Farbe grau ist.

Mammal $\sqcap \exists bodypart. Trunk \sqcap \forall color. Grey$

Manchester-Syntax:

Mammal **and** bodypart **some** Trunk **and** color **only** Grey

ALC: Sätze

Definition

Die Menge $\text{Sen}(\Sigma)$ der \mathcal{ALC} -Sätze über Σ wird definiert als

- Subsumptionen $C \sqsubseteq D$, wobei $C, D \in \mathbf{Con}(\Sigma)$
Class: C **SubclassOf:** D
- Konzept-Mitgliedschaften $a : C$, wobei $a \in I$ und $C \in \mathbf{Con}(\Sigma)$
Individual: a **Types:** C
- Rollen-Mitgliedschaften $R(a_1, a_2)$, wobei $R \in \mathbf{R}$ und $a_1, a_2 \in I$
Individual: a1 **Facts:** R a2

$C \equiv D$ ist eine Abkürzung für die beiden Sätze

- $C \sqsubseteq D$
- $D \sqsubseteq C$

Manchester-Syntax: **Class:** C **EquivalentTo:** D

ALC: Beispielsatz

Ein **Elefant** ist ein **graues Säugetier** mit einem **Rüssel**.

Ein **Elefant** ist ein **Säugetier**, das einen **Rüssel** als **Körperteil** hat und dessen **einige Farbe grau** ist.

Elephant ≡ Mammal ⊓ ∃bodypart.Trunk ⊓ ∀color.Grey

Manchester-Syntax:

Class: Elephant EquivalentTo:

Mammal **and** bodypart **some** Trunk **and** color **only** Grey

Formalisierungsbeispiele

- Eltern haben Kinder

$\text{Parent} \sqsubseteq \exists \text{hasChild.} \top$

$\text{Parent} \sqsubseteq \exists \text{hasChild.} \text{Child}$

- Kinder haben Eltern

$\text{Child} \sqsubseteq \exists \text{hasParent.} \top$

$\text{Child} \sqsubseteq \exists \text{hasParent.} \text{Parent}$

- Ein Großvater ist ein Vater, der ein Elternteil zum Kind hat

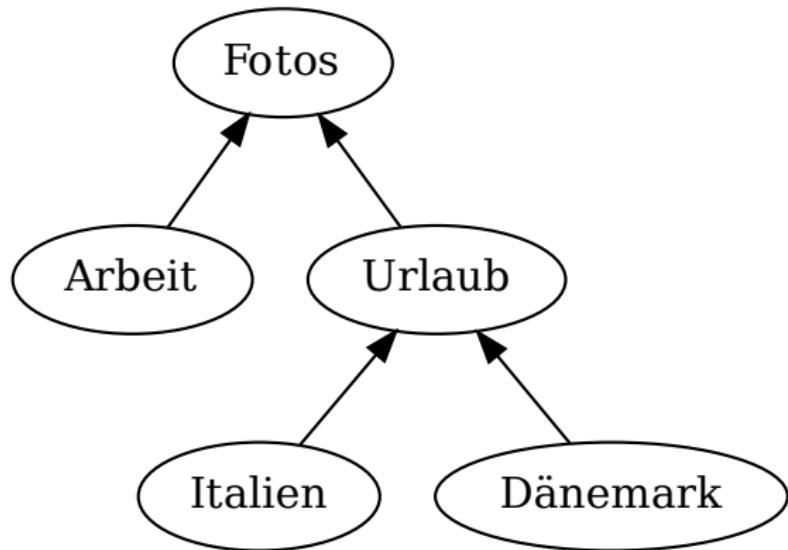
$\text{Grandfather} \equiv \text{Father} \sqcap (\exists \text{hasChild.} \text{Parent})$

TBoxen und ABoxen

Logische Theorien in Beschreibungslogiken werden üblicherweise in zwei Teile aufgeteilt:

- **TBox**: Subsumptionen und Definitionen für Konzepte
 - $Woman \sqsubseteq Person$
class: Woman **SubClassOf:** Person
- **ABox**: Individuen und ihre Mitgliedschaft in Konzepten und Rollen
 - $john : Father, hasChild(john, harry)$
Individual: john **Types:** Father **Facts:** hasChild harry

Kein sinnvolles Beispiel für eine Taxonomie / TBox



Beispiel-TBox: die Open Energy Ontology

- basiert auf der Upper Ontology BFO
- formalisiert Begriffe in der Domäne Energiesystemmodellierung
- Homepage
<https://openenergy-platform.org/ontology>
- Repository
<https://github.com/OpenEnergyPlatform/ontology>
- OWL2-Theorie <https://raw.githubusercontent.com/OpenEnergyPlatform/ontology/dev/src/ontology/oeo.owl>
 - zum Laden in den Ontologie-Editor Protégé

Semantik

Semantik von ALC

Wir definieren eine Übersetzungssemantik für \mathcal{ALC} :

- \mathcal{ALC} -Signaturen werden in PL1-Sprachen übersetzt
 - \mathcal{ALC} -Sätze werden in PL1-Sätze übersetzt
 - dies setzt sich auf logische Theorien (Mengen von Sätzen) fort
 - \mathcal{ALC} -Strukturen sind dann PL1-Strukturen für die übersetzte Signatur
 - ein \mathcal{ALC} -Satz gilt in einer \mathcal{ALC} -Struktur, wenn seine Übersetzung in einen PL1-Satz in der Struktur gilt

Übersetzung von ALC nach PL1: Signaturen

Eine \mathcal{ALC} -Signatur $(\mathbf{C}, \mathbf{R}, \mathbf{I})$ wird in folgende PL1-Sprache übersetzt:

Individuenkonstanten: alle Individuenkonstanten aus \mathbf{I}

einstellige Prädikatsymbole: alle Konzepte aus \mathbf{C}

zweistellige Prädikatsymbole: alle Rollen aus \mathbf{R}

Beispiel: Signatur der Familien-Ontologie

In \mathcal{ALC} :

- $C = \{Person, Woman, Father, Male, \dots\}$
- $R = \{hasChild, hasParent, brotherOf, auntOf, \dots\}$
- $I = \{\text{peter}, \text{mary}, \text{john}, \dots\}$

Übersetzung nach PL1:

- einstellige Prädikatsymbole:
 $\{Person, Woman, Father, Male, \dots\}$
- zweistellige Prädikatsymbole:
 $\{hasChild, hasParent, brotherOf, auntOf, \dots\}$
- Individuenkonstanten: $\{\text{peter}, \text{mary}, \text{john}, \dots\}$

Übersetzung von ALC nach PL1: Konzepte

\mathcal{ALC} -Konzepte C werden in wohlgeformte PL1-Formeln $\alpha_x(C)$ mit einer freien Variablen x übersetzt:

- $\alpha_x(\perp) = \perp$
- $\alpha_x(\top) = \neg\perp$
- $\alpha_x(A) = A(x)$
- $\alpha_x(\neg C) = \neg\alpha_x(C)$
- $\alpha_x(C \sqcap D) = \alpha_x(C) \wedge \alpha_x(D)$
- $\alpha_x(C \sqcup D) = \alpha_x(C) \vee \alpha_x(D)$
- $\alpha_x(\exists R.C) = \exists y(R(x, y) \wedge \alpha_y(C))$ (y ist eine neue Variable)
- $\alpha_x(\forall R.C) = \forall y(R(x, y) \rightarrow \alpha_y(C))$ (y ist eine neue Variable)

ALC: Beispiel einer Übersetzung eines Konzepts

Ein Säugetier, das einen Rüssel als Körperteil hat und dessen einzige Farbe grau ist.

Mammal $\sqcap \exists bodypart. Trunk \sqcap \forall color. Grey$

Manchester-Syntax:

Mammal **and** bodypart **some** Trunk **and** color **only** Grey

Übersetzung in wohlgeformte PL1-Formel mit freier Variablen x:

$Mammal(x) \wedge \exists y (bodypart(x, y) \wedge Trunk(y))$
 $\wedge \forall z (color(x, z) \rightarrow Grey(z))$

Übersetzung von ALC nach PL1: Sätze

\mathcal{ALC} -Sätze werden in PL1-Sätze übersetzt (unter Verwendung der Konzeptübersetzung der vorigen Folie):

- $\alpha(C \sqsubseteq D) = \forall x(\alpha_x(C) \rightarrow \alpha_x(D))$
- $\alpha(a : C) = \alpha_x(C)[x \mapsto a]$
- $\alpha(R(a, b)) = R(a, b)$

ALC: Übersetzung eines Beispielsatzes

Ein Elefant ist ein Säugetier, das einen Rüssel als Körperteil hat und dessen einzige Farbe grau ist.

Elephant \equiv *Mammal* $\sqcap \exists \text{bodypart}. \text{Trunk} \sqcap \forall \text{color}. \text{Grey}$

Manchester-Syntax:

Class: Elephant **EquivalentTo:**

Mammal and bodypart some Trunk and color only Grey

Übersetzung in einen PL1-Satz:

$$\forall x [Elephant(x) \leftrightarrow (Mammal(x) \wedge \exists y (bodypart(x, y) \wedge Trunk(y)) \wedge \forall z (color(x, z) \rightarrow Grey(z)))]$$

Schlussfolgern

TBox-Reasoning

TBox-Reasoning fragt, ob eine Subsumption aus einer TBox folgt:

?

$$\mathcal{T} \models_{PL1} C \sqsubseteq D$$

Beispiel: Pizza-Ontologie \mathcal{T}

VegetarianPizza	\sqsubseteq	Pizza
MagheritaPizza	\sqsubseteq	Pizza
TomatoTopping	\sqsubseteq	VegetableTopping
MozzarellaTopping	\sqsubseteq	CheeseTopping
VegetarianPizza	\equiv	$\forall \text{ hasTopping } (\text{VegetableTopping} \sqcup \text{CheeseTopping})$
MagheritaPizza	\sqsubseteq	$\exists \text{ hasTopping } \text{MozarellaTopping} \sqcap$ $\exists \text{ hasTopping } \text{TomatoTopping} \sqcap$ $\forall \text{ hasTopping }$ $(\text{MozarellaTopping} \sqcup \text{TomatoTopping})$

Logische Folgerung: $\mathcal{T} \models_{PL1} \text{MagheritaPizza} \sqsubseteq \text{VegetarianPizza}$

TBox-Reasoning und Unerfüllbarkeit

Üblicherweise wird **Unerfüllbarkeit** von Konzepten getestet:

Definition

Ein \mathcal{ALC} -Konzept C ist **unerfüllbar** bzgl. einer TBox \mathcal{T} , in Zeichen $\mathcal{T} \models_{PL1} \text{unsat}(C)$, wenn es keine PL1-Struktur für \mathcal{T} gibt, in der die Extension von C nicht-leer ist.

Unerfüllbarkeit und Subsumption können gegenseitig aufeinander reduziert werden:

$$\begin{array}{lll} \mathcal{T} \models_{PL1} C \sqsubseteq D & \text{genau dann, wenn} & \mathcal{T} \models_{PL1} \text{unsat}(C \sqcap \neg D) \\ \mathcal{T} \models_{PL1} \text{unsat}(C) & \text{genau dann, wenn} & \mathcal{T} \models_{PL1} C \sqsubseteq \perp \end{array}$$

Für diese Reasoning-Probleme sind effiziente Beweistools verfügbar:
Konclude, Pellet, Fact++, Racer.

Beispiel: Unerfüllbarkeit

Beispiel: Spezialisierung \mathcal{T} der Pizza-Ontologie

VegetarianPizza	\sqsubseteq	Pizza
MeatPizza	\sqsubseteq	Pizza
VegetarianPizza \sqcap MeatPizza	\sqsubseteq	\perp
MyFancyPizza	\sqsubseteq	VegetarianPizza
MyFancyPizza	\sqsubseteq	MeatPizza

Logische Folgerung: $\mathcal{T} \models_{PL1} \text{unsat}(\text{MyFancyPizza})$

sROIQ und OWL2 DL

Die Logik sROIQ

Die Logik *sROIQ* ist eine Erweiterung von *ALC*. Sie kennt noch weitere Formen von Sätzen:

- Kardinalitäts-Restriktionen:

Kinder haben genau zwei Eltern:

Child $\sqsubseteq = 2 \text{ hasParent. Parent}$

- Inverse von Rollen: $\text{Child} \sqsubseteq \exists \text{hasChild}^- . \top$

- Transitivität von Rollen: $\text{Transitive}(\text{ancestorOf})$

- ...

Die Ontologiesprache OWL 2 DL basiert auf *sROIQ* und hat darüber hinaus eine **Web-kompatible Syntax** mit URLs bzw. IRIs für alle Bezeichner.

Zusammenfassung

Veranstaltung 14

Zusammenfassung Veranstaltung 14 und Ausblick

Was haben wir heute gelernt?

- die **Ontologiesprache *ALC***
 - gleiche Semantik wie PL1
 - andere Syntax
 - intuitiver
 - variablenfrei

Wie geht es weiter?

- Logik II
- Grundlagen **semantischer Technologien**
- Einführung in die Angewandte **Ontologie**
- **Automated Reasoning**
- Grundlagen der Theoretischen Informatik I+II