Formale Grundlagen der Informatik 1
Kapitel 14
Aussagenlogik
Syntax & Semantik

Frank Heitmann
heitmann@informatik.uni-hamburg.de

1. Juni 2015

Motivation

Mit der Aussagenlogik lassen sich einfache Verknüpfungen zwischen (atomaren) Gebilden ausdrücken z.B.

- $A \wedge B$ für A *und* B
- $A \vee B$ für A *oder* B

Wenn A und B für etwas stehen (z.B. $A \approx$ 'es regnet') lassen sich so kompliziertere Aussagen formen.

Motivation

Man kann dann

1. Etwas aus der realen Welt in der Logik abstrakt ausdrücken.
2. In der Logik Schlüsse ziehen.
3. Dies wieder in der realen Welt interpretieren.

Motivation

(Aussagen-)Logik ...

- als Grundlage der Mathematik,
- für Programmiersprachen (z.B. Prolog),
- für künstliche Intelligenzen,
- für Datenbanken,
- zur Beschreibung von Schaltkreisen,
- in der Verifikation
- ...

Motivation

Die **Aussagenlogik**

- ist eine ganz grundlegende Logik (Basis vieler anderer Logiken bzw. in ihnen enthalten)
- an ihr lässt sich vieles einüben
- ist uns schon im SAT-Problem begegnet (und ist also ganz grundlegend für den Begriff der NP-Vollständigkeit und der Frage, was effizient lösbar ist)

Motivation

- Eine *Aussage* im Sinne der Aussagenlogik ist ein atomares sprachliches Gebilde das entweder *wahr* oder *falsch* ist. Notiert als A, B, C oder A_1, A_2, A_3, ... Diese nennt man *Aussagensymbole*.
- Die *Aussagenlogik* betrachtet den Wahrheitsgehalt einfacher Verknüpfungen zwischen atomaren sprachlichen Gebilden (also Aussagen). Dies sind:
  - $\neg$ für *nicht* (Negation)
  - $\wedge$ für *und* (Konjunktion)
  - $\vee$ für *oder* (Disjunktion)
  - $\Rightarrow$ für *wenn ... dann* (Implikation)
  - $\Leftrightarrow$ für *genau dann, wenn* (Biimplikation)

  Die $\neg, \wedge, \vee, \Rightarrow, \Leftrightarrow$ nennt man Junktoren.

Exkurs

Belegt man einzelne Aussagensymbole durch einfache (atomare) Aussagen, so kann man (natürlichsprachliche) Sätze übertragen. Z.B.

- $A =$ es regnet
- $B =$ ich trage einen Schirm
- $A \Rightarrow B =$ Wenn es regnet, dann trage ich einen Schirm

(Dabei ist aber Vorsicht geboten, da das nicht immer dem entspricht, was man intuitiv von der natürlichen Sprache erwartet.)

Syntax: Motivation

Die **Syntax** legt nun zunächst nur fest, wie mit *atomaren Formeln* und *Junktoren* komplexe *Formeln* erstellen kann. Diese Formeln sind zunächst nur Zeichenkette ohne Bedeutung (Semantik).

Syntax: Definition

Definition (Syntax der Aussagenlogik)

Mit AS_{AL} sei die Menge der *Aussagensymbol* der Aussagenlogik bezeichnet. Wir notieren diese üblicherweise als $A_1, A_2, A_3, \ldots$ oder $A, B, C, \ldots$.
Die Menge $\mathcal{L}_{AL}$ der Formeln der Aussagenlogik definieren wir mittels

1. Jedes $A \in AS_{AL}$ ist eine (atomare) Formel.
2. Ist F eine Formel, so ist auch $\neg F$ eine Formel.
3. Sind F und G Formeln, so sind auch $(F \vee G), (F \wedge G), (F \Rightarrow G)$ und $(F \Leftrightarrow G)$ Formeln.
4. Es gibt keine anderen Formeln, als die, die durch *endliche Anwendungen* der Schritte 1-3 erzeugt werden.

Syntax: Definition

Noch ein paar Bezeichnungen:

- Manchmal führt man noch das **Alphabet** ein. Dies besteht aus den Aussagesymbolen sowie aus den Junktoren und den Klammern (und).
- Die $\neg, \vee, \wedge, \Rightarrow, \Leftrightarrow$ werden als **Junktoren** bezeichnet. Die entstehenden Formeln als Negation ($\neg$), Disjunktion ($\vee$), Konjunktion ($\wedge$), Implikation ($\Rightarrow$) und Biimplikation ($\Leftrightarrow$).
- Eine Formel, die beim Aufbau einer Formel F verwendet wird, heißt **Teilformel** von F. Außerdem ist F Teilformel von sich selbst.
- Der Junktor, der im letzten Konstruktionsschritt verwendet wird, heißt **Hauptoperator**. Nach ihm werden komplexe Formeln benannt.

Syntax: Beispiel

Beispiele:

- $((A \vee C) \wedge B)$. Dies ist eine Konjunktion, da zuletzt $\wedge$ angewandt wurde. Teilformeln sind $A, B, C, (A \vee C)$ und $((A \vee C) \wedge B)$.
- $(A \vee \vee C)$ ist keine Formel.
- $A \vee C$ zunächst auch nicht (Klammerung!)

Strukturbäume

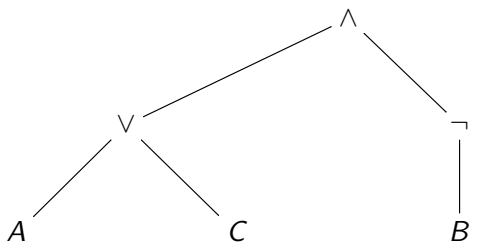
Man kann Formeln auch durch sogenannte Strukturbäume ausdrücken:

- Der Hauptoperator markiert die Wurzel.
- Teilformeln entsprechen Teilbäumen.
- Die Reihenfolge der Teilformeln wird beibehalten (linke Teilformel, linkes Kind).

Die Aussagesymbole werden dann die Blätter des Baumes, die Junktoren sind die inneren Knoten. Dabei hat $\neg$ ein Kind, alle anderen Junktoren zwei.

Strukturbäume

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



Strukturelle Induktion und Rekursion

Den Aufbau komplexer Formeln aus einfache(re)n Formeln kann man nutzen um

1. Eigenschafen von Formeln nachzuweisen (strukturelle Induktion)
2. Funktionen über die Formelmenge zu definieren (strukturelle Rekursion)

Strukturelle Induktion

Um eine Behauptung $B(F)$ für jede Formel $F \in \mathcal{L}_{AL}$ zu zeigen genügt es:

1. Zu zeigen, dass $B(F)$ für jede atomare Formel F gilt (**Induktionsanfang**).
2. Anzunehmen, dass $B(F)$ und $B(G)$ für zwei Formeln F und G gilt (**Induktionsannahme**).
3. Zu zeigen, dass unter der Annahme bei 2. nun auch $B(\neg F), B(F \vee G), B(F \wedge G), B(F \Rightarrow G)$ und $B(F \Leftrightarrow G)$ gelten (**Induktionsschritt**).

Strukturelle Rekursion

Um eine Funktion $f : \mathcal{L}_{AL} \to D$ zu definieren (D ist dabei eine beliebige Menge) genügt es:

1. $f(A)$ für jedes $A \in AS_{AL}$ festzulegen.
2. eine Funktion $f_\neg : D \to D$ und für jeden Junktor $\circ \in \{\vee, \wedge, \Rightarrow, \Leftrightarrow\}$ eine Funktion $f_\circ : D \times D \to D$ zu definieren. Es ist dann z.B. $f((F \wedge G)) = f_\wedge(f(F), f(G))$.

Strukturelle Rekursion: Beispiele

Wir definieren **Grad** und **Tiefe** einer Formel:

1. Für jedes $A \in AS_{AL}$ sei
   - $grad(A) = tiefe(A) = 0$.
2. Für $\neg$ sei
   - $grad(\neg F) = grad_{\neg}(grad(F)) = grad(F) + 1$ und
   - $tiefe(\neg F) = tiefe_{\neg}(tiefe(F)) = tiefe(F) + 1$.
3. Für $\circ \in \{\vee, \wedge, \Rightarrow, \Leftrightarrow\}$ sei
   - $grad((F \circ G)) = grad_{\circ}(grad(F), grad(G)) = grad(F) + grad(G) + 1$ und
   - $tiefe((F \circ G)) = tiefe_{\circ}(tiefe(F), tiefe(G)) = max(tiefe(F), tiefe(G)) + 1$.

Grad und Tiefe

Die Funktion *grad* zählt die Anzahl der Junktoren. Die *tiefe* zählt die Anzahl der Junktoren auf dem längsten Pfad im Strukturbaum (also quasi die tiefste Verschachtelung).

Anmerkung

Man definiert also die **Rekursionsbasis** (was bei den Aussagesymbolen passiert) und den **Rekursionsschritt** (was bei den einzelnen Junktoren passiert; wobei man hier Negation und die zweistelligen Junktoren meist getrennt behandelt. Die zweistelligen Junktoren müssen dabei nicht wie oben alle gleich behandelt werden.)

Zusammenfassung

Zusammenfassung bis hier:

- Motivation
- Definition der **Syntax**
  - Alphabet, Junktor
  - Aussagesymbol, atomare Formel, komplexe Formel
  - Hauptoperator, Teilformel
  - Negation, Disjunktion, Konjunktion, Implikation, Biimplikation
- Strukturbäume
- strukturelle Induktion
- strukturelle Rekursion
- Grad und Tiefe einer Formel

Grundbegriffe

Wir wollen nun die *Bedeutung* von Formeln definieren.

Dazu

- **belegen** wir die atomaren Formeln mit **Wahrheitswerten**
- berechnen daraus den Wahrheitswert einer komplexen Formel

Die Menge der Wahrheitswerte enthält genau zwei Elemente

- 1 ('wahr') und
- 0 ('falsch').

Syntax: Definition (Wdh.)

Definition (Syntax der Aussagenlogik)

Mit AS_{AL} sei die Menge der *Aussagensymbol* der Aussagenlogik bezeichnet. Wir notieren diese üblicherweise als $A_1, A_2, A_3, \ldots$ oder $A, B, C, \ldots$.
Die Menge $\mathcal{L}_{AL}$ der Formeln der Aussagenlogik definieren wir mittels

1. Jedes $A \in AS_{AL}$ ist eine (atomare) Formel.
2. Ist F eine Formel, so ist auch $\neg F$ eine Formel.
3. Sind F und G Formeln, so sind auch $(F \vee G), (F \wedge G), (F \Rightarrow G)$ und $(F \Leftrightarrow G)$ Formeln.
4. Es gibt keine anderen Formeln, als die, die durch *endliche Anwendungen* der Schritte 1-3 erzeugt werden.

Semantik

- Eine *Belegung* weist nun jedem Aussagesymbol einen Wahrheitswert zu.
- Aussagen und Formeln können dann unter einer Belegung wahr oder falsch sein.
- Die aussagenlogische Semantik regelt u.a., wie komplexe Formeln zu Wahrheitswerten kommen.

Semantik

Definition (Semantik der Aussagenlogik)

Eine **Belegung** ist eine Funktion $\mathcal{A}_{AS} : AS_{AL} \to \{0, 1\}$, die jedem Aussagesymbol einen Wahrheitswert zuordnet.
Zu dieser wird rekursiv eine Funktion $\mathcal{A} : \mathcal{L}_{AL} \to \{0, 1\}$ definiert, die alle Formeln bewertet. Es ist für jedes $A \in AS_{AL}$ ist $\mathcal{A}(A) = \mathcal{A}_A S(A)$ und für alle Formeln $F, G \in \mathcal{L}_{AL}$ sei

- $\mathcal{A}(\neg F) = 1$ genau dann, wenn $\mathcal{A}(F) = 0$
- $\mathcal{A}((F \vee G)) = 1$ gdw. $\mathcal{A}(F) = 1$ oder $\mathcal{A}(G) = 1$
- $\mathcal{A}((F \wedge G)) = 1$ gdw. $\mathcal{A}(F) = 1$ und $\mathcal{A}(G) = 1$
- $\mathcal{A}((F \Rightarrow G)) = 1$ gdw. $\mathcal{A}(F) = 0$ oder $\mathcal{A}(G) = 1$
- $\mathcal{A}((F \Leftrightarrow G)) = 1$ gdw. $\mathcal{A}(F) = \mathcal{A}(G)$

Semantik - Anmerkung

Anmerkung

Bspw. die Definition

$$\mathcal{A}((F \vee G)) = 1 \text{ gdw. } \mathcal{A}(F) = 1 \text{ oder } \mathcal{A}(G) = 1$$

bedeutet, dass $\mathcal{A}(F \vee G)$ zu 1 ausgewertet wird, wenn

- $\mathcal{A}(F) = 1$ ist oder wenn
- $\mathcal{A}(G) = 1$ ist oder wenn
- beides gilt.

In allen anderen Fällen (hier nur $\mathcal{A}(F) = \mathcal{A}(G) = 0$) ist $\mathcal{A}((F \vee G)) = 0$.

Semantik - Wahrheitstafeln

Wahrheitstafeln geben für die atomaren Formeln alle möglichen Belegungen an und für die anderen Formeln die entsprechenden Bewertungen. Sie stellen die Definition von eben übersichtlich dar.

A	B	$\neg A$	$A \vee B$	$A \wedge B$	$A \Rightarrow B$	$A \Leftrightarrow B$
0	0	1	0	0	1	1
0	1	1	1	0	1	0
1	0	0	1	0	0	0
1	1	0	1	1	1	1

Zur Nachbereitung

Für $\neg A$ hätte auch die kleinere Tabelle

A	$\neg A$
0	1
1	0

genügt, aber so wie oben hat dann alles in eine Tabelle gepasst.

Aufgabe

A	B	C	D	$A \vee \neg B$	$C \wedge \neg D$	$\neg(A \vee \neg B) \wedge (C \wedge \neg D)$
0	0	0	0			
0	0	0	1			
0	0	1	0			
0	0	1	1			
0	1	0	0			
0	1	0	1			
0	1	1	0			
0	1	1	1			
1	0	0	0			
1	0	0	1			
1	0	1	0			
1	0	1	1			
1	1	0	0			
1	1	0	1			
1	1	1	0			
1	1	1	1			

Lösung der Aufgabe

A	B	C	D	$A \vee \neg B$	$C \wedge \neg D$	$\neg(A \vee \neg B) \wedge (C \wedge \neg D)$
0	0	0	0	1	0	0
0	0	0	1	1	0	0
0	0	1	0	1	1	0
0	0	1	1	1	0	0
0	1	0	0	0	0	0
0	1	0	1	0	0	0
0	1	1	0	0	1	1
0	1	1	1	0	0	0
1	0	0	0	1	0	0
1	0	0	1	1	0	0
1	0	1	0	1	1	0
1	0	1	1	1	0	0
1	1	0	0	1	0	0
1	1	0	1	1	0	0
1	1	1	0	1	1	0
1	1	1	1	1	0	0

Wahrheitstafeln: Anmerkungen

- In jeder Zeile einer Wahrheitstafel steht eine Belegung.
- Jede Zeile beschreibt einen (prinzipiell) möglichen Zustand der Welt.
- Enthält eine Formel n verschiedene atomare Formeln / Aussagensymbole, so existieren 2^n Zeilen in der Tafel.
- Eine Spalte wird als **Wahrheitswerteverlauf (WWV)** der zugehörigen Formel bezeichnet.

Kategorien

Definition

- Eine Belegung $\mathcal{A}$ mit $\mathcal{A}(F) = 1$ nennt man ein **Modell** für F oder eine *erfüllende Belegung* von F. Ist $\mathcal{A}(F) = 0$, so ist $\mathcal{A}$ eine *falsifizierende Belegung* von F.
- Ist ferner M eine (evtl. sogar unendliche) Formelmenge. So nennt man eine Belegung $\mathcal{A}$, die alle Formeln F aus M wahr macht, ebenfalls ein *Modell* für M und schreibt dafür bisweilen auch kurz $\mathcal{A}(M) = 1$.
- Zudem ist jede Belegung Modell der leeren Menge. Die leere Menge ist also erfüllbar.

Kategorien

Definition

- Besitzt F mindestens eine erfüllende Belegung (ein Modell), so heißt F **erfüllbare** Formel.
- Besitzt F mindestens eine falsifizierende Belegung, so heißt F **falsifizierbare** Formel.
- Besitzt F mindestens eine erfüllende und mindestens eine falsifizierende Belegung so heißt F **kontingente** Formel.
- Besitzt F kein Modell, so heißt F **unerfüllbare** Formel oder **Kontradiktion**.
- Ist F unter jeder möglichen Belegung „wahr“, so heißt F **(allgemein-)gültig** oder **Tautologie**.

Kategorien - Notationen

Notationen:

- $\mathcal{A}$ ist Modell von F bzw. macht F wahr wird kurz geschrieben als $\mathcal{A} \models F$.
- $\mathcal{A}$ falsifiziert F bzw. macht F falsch wird kurz geschrieben als $\mathcal{A} \not\models F$.
- Ist F eine Tautologie, wird dies kurz notiert als $\models F$.
- Ist F eine Kontradiktion, wird dies kurz notiert als $F \models$.

Tautologie vs. Kontradiktion

Satz

F ist gültig genau dann, wenn $\neg F$ unerfüllbar ist.

Beweis.

	F ist gültig	
gdw.	jede Belegung ist ein Modell von F	(Def. der Gültigkeit)
gdw.	$\mathcal{A}(F) = 1$ für jede Belegung $\mathcal{A}$	(Def. eines Modells)
gdw.	$\mathcal{A}(\neg F) = 0$ für jede Belegung $\mathcal{A}$	(Eigenschaft von $\neg$)
gdw.	keine Belegung ist ein Modell von $\neg F$	(Def. eines Modells)
gdw.	$\neg F$ ist unerfüllbar	(Def. der Unerfüllbarkeit)

□

Zusammenfassung 1

Zusammenfassung **Syntax**:

- Motivation
- Definition der Syntax:
  - Alphabet, Junktor
  - Aussagesymbol, atomare Formel, komplexe Formel
  - Hauptoperator, Teilformel
  - Negation, Disjunktion, Konjunktion, Implikation, Biimplikation
- Strukturbäume
- strukturelle Induktion
- strukturelle Rekursion
- Grad und Tiefe einer Formel

Zusammenfassung 2

Zusammenfassung **Semantik**:

- Belegung, Auswertung (einer Formel)
- Wahrheitstafeln, Wahrheitswerteverlauf
- erfüllende Belegung, falsifizierende Belegung, Modell
- kontingent, (allgemein-)gültig, unerfüllbar
- Tautologie, Kontradiktion
- $\mathcal{A} \models F$, $\mathcal{A} \not\models F$, $\models F$, $F \models$

Ausblick

Nächstes Mal beschäftigen wir uns mit

- äquivalenten Formeln,
- (Äquivalenz-)Umformungen und
- der Herstellung zweier *Normalformen*.